

IAP20 RECEIVED 22 MAR 2006

明細書

再生装置、プログラム、再生方法

技術分野 本発明は、デジタル化された映画作品の再生と、アプリケーションの
5 5 ンの実行とを同時に実行する、再生制御技術の技術分野に属する発明であり、
本再生制御技術を民生用の再生装置、プログラムに応用する場合の応用技術に
深く係る。

背景技術

10 映画ビジネスには、デジタル化された映画作品と、ゲームアプリケーション
とを1つのパッケージに収めて販売するというメディアミックスの形態がある。
このゲームアプリケーションが、ゲームの登場人物をキャラクタにしており、
デジタル化された映画作品の一部分と同時に実行されるのなら、映像再生と、
ゲーム実行との双方による相乗効果で、映画作品の人気は一層高まる。
15 しかしながらアプリケーションには、程度の差はあれ、多かれ少なかれバグ
がある。バグが潜在していたため、装置がブラックアウトするのは、パソコン
ソフトの世界では"よくあること"で済まされるかもしれないが、民生機器の
分野では品質問題になりかねない。品質問題への波及がありうるので、メーカ
各社は、アプリケーションの実行と、デジタルストリーム再生とを同時に実行
20 20 するような再生装置の実現に二の足を踏むことが多かった(注:ブラックアウト
とは、装置のソフトウェアがフリーズして表示画面が真っ暗になる状態をいう)。

発明の開示

本発明の目的は、品質問題への波及を避けつつ、アプリケーションの実行と、
25 25 デジタルストリーム再生とを同時に実行することができる再生装置を提供する
ことである。

上記目的は、タイトルの再生と、アプリケーションの実行とを行う再生装置
であって、タイトルに帰属するデジタルストリームを再生する再生制御エンジ
ン部と、複数タイトル間の分岐を制御するモジュールマネージャと、1つ以上

のアプリケーションを実行するモジュールとを備え、前記モジュールは、仮想マシン部と、アプリケーションマネージャを含み、前記アプリケーションマネージャは、1 つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈して終了処理を行い、前記モジュールマネージャは、
5 前記タイトルの終了後、所定のタイトルを選択する、再生装置により達成される。

アプリケーションを含むが、デジタルストリームは含まないようなタイトルの終了時においても、所定のタイトルに分岐するという制御が可能になる。これによりアプリケーションプログラムがエラー終了したり、アプリケーション
10 の起動に失敗したとしても、違和感がない制御を保証することができる。

こうした保証があるので、品質問題への波及に神経を尖らせているメーカ各社に対しても、ストリーム再生と、アプリケーション実行とを同時に実行する再生装置の開発を強く後押しすることができる。こうした強い後押しにより、再生装置の低価格化、多様化が進めば、BD-ROM コンテンツの充実化を図る
15 ことができるので、コンテンツ関連産業の発達を強く牽引することができる。

図面の簡単な説明

図 1 は、本発明に係る再生装置の使用行為についての形態を示す図である。

図 2 は、BD-ROM におけるファイル・ディレクトリ構成を示す図である。

20 図 3 は、AVClip 時間軸と、PL 時間軸との関係を示す図である。

図 4 は、4 つの Clip_Information_file_name よりなされた一括指定を示す図である。

図 5 は、PLmark によるチャプター定義を示す図である。

25 図 6 は、SubPlayItem 時間軸上の再生区間定義と、同期指定とを示す図である。

図 7 (a) は、Movie オブジェクトの内部構成を示す図である。

図 7 (b) は、BD-J オブジェクトの内部構成を示す図である。

図 7 (c) は、Java アプリケーションの内部構成を示す図である。

図 8 (a) は、Java アーカイブファイルに収められているプログラム、デー

タを示す図である。

図8 (b) は、xlet プログラムの一例を示す図である。

図9 (a) は、トップメニュー、title#1、title#2 といった一連のタイトルを示す図である。

5 図9 (b) は、PlayList#1、PlayList#2 の時間軸を足し合わせた時間軸を示す図である。

図10 は、本編タイトル、オンラインショッピングタイトル、ゲームタイトルという3つのタイトルを含むディスクコンテンツを示す図である。

図11 は、図10 に示した3つのタイトルの再生画像の一例を示す図である。

10 図12 (a) は、図10 の破線に示される帰属関係から各アプリケーションの生存区間をグラフ化した図である。

図12 (b) は、図12 (a) の生存区間を規定するため、記述されたアプリケーション管理テーブルの一例を示す図である。

図13 (a) は、起動属性設定の一例を示す図である。

15 図13 (b) は、他のアプリケーションからのアプリケーション呼出があって初めて起動するアプリケーション(application#2)を示す図である。

図14 (a) (b) は、Suspend が有意義となるアプリケーション管理テーブル、生存区間の一例を示す図である。

図15 は、起動属性がとり得る三態様(Persistent、AutoRun、Suspend)と、
20 直前タイトルにおけるアプリケーション状態の三態様(非起動、起動中、Suspend)とがとりうる組合せを示す図である。

図16 は、本発明に係る再生装置の内部構成を示す図である。

図17 (a) は、BD-ROM に存在している Java アーカイブファイルを、ローカルメモリ29上でどのように識別するかを示す図である。

25 図17 (b) は、図17 (a) の応用を示す図である。

図18 は、ROM24に格納されたソフトウェアと、ハードウェアとからなる部分を、レイア構成に置き換えて描いた図である。

図19 は、Presentation Engine 31～モジュールマネージャ34による処理を模式化した図である。

図20は、アプリケーションマネージャ36による処理を模式化した図である。

図21は、ワークメモリ37～Default Operation Manager 40を示す図である。

- 5 図22は、アプリケーションマネージャ36による分岐時の制御手順を示す図である。

図23は、アプリケーション終了処理の処理手順を示すフローチャートである。

図24は、アプリケーション終了の過程を模式的に示した図である。

- 10 図25(a)は、PL 時間軸上に生存区間を定めたアプリケーション管理テーブルを示す図である。

図25(b)は、図25(a)のアプリケーション管理テーブルに基づき、アプリケーションの生存区間を示した図である。

図26(a)は、PL 時間軸から定まるタイトル時間軸を示す。

- 15 図26(b)は、メインとなるアプリケーションの生存区間から定まるタイトル時間軸を示す。

図26(c)は、複数アプリケーションの生存区間から定まるタイトル時間軸を示す図である。

- 20 図27は、タイトル再生時におけるアプリケーションマネージャ36の処理手順を示すフローチャートである。

図28(a)は、BD-ROMにより実現されるメニュー階層を示す図である。

図28(b)は、メニュー階層を実現するためのMOVIEオブジェクトを示す図である。

- 25 図29は、Index Table と、Index Table から各 Movie オブジェクトへの分岐とを模式化した図である。

図30(a)は、図29(b)のようにIndex Table が記述された場合における分岐を示す。

図30(b)は、非AV系タイトルが強制終了した際における分岐を示す図である。

図31は、モジュールマネージャ34の処理手順を示すフローチャートである。

図32は、アプリケーションマネージャ36によるアプリケーション強制終了の動作例を示す図である。

5 図33は、Playback Control Engine 32によるPL再生手順を示すフローチャートである。

図34は、アングル切換、SkipBack, SkipNextの受付手順を示すフローチャートである。

10 図35は、SkipBack, SkipNextAPIがコールされた際の処理手順を示すフローチャートである。

図36は、Presentation Engine 31による処理手順の詳細を示すフローチャートである。

図37は、SubPlayItemの再生手順を示すフローチャートである。

15 図38は、第5実施形態に係るアプリケーションマネージャ36の処理手順を示すフローチャートである。

図39は、データ管理テーブルの一例を示す図である。

図40は、BD-Jオブジェクトが想定している実行モデルを示す図である。

図41(a)は、ローカルメモリ29におけるJavaアーカイブファイル生存を示す生存区間を示す図である。

20 図41(b)は、図41(a)でのJavaアーカイブファイル生存区間を規定するため、記述されたデータ管理テーブルを示す図である。

図42は、カルーセル化によるJavaアーカイブファイル埋め込みを示す図である。

図43(a)は、インターリーブ化によるAVClip埋め込みを示す図である。

25 図43(b)は、読込属性の3つの類型を示す図である。

図44(a)は、データ管理テーブルの一例を示す図である。

図44(b)は、図44(a)のデータ管理テーブルの割り当てによるローカルメモリ29の格納内容の変遷を示す図である。

図45(a)は、新旧再生装置におけるローカルメモリ29のメモリ規模を

対比して示す図である。

図45 (b) は、読込優先度が設定されたデータ管理テーブルの一例を示す図である。

図46 は、アプリケーションマネージャ36によるプリロード制御の処理手順を示す図である。

図47 (a) は、applicationID が同一であるが、読込優先度は互いに異なる複数のアプリケーションを規定するデータ管理テーブルの一例を示す図である。

図47 (b) は、図47 (a) のデータ管理テーブルの割り当てによるローカルメモリ29の格納内容の変遷を示す図である。

図48 (a) は、プリロードされるべきアプリケーション、ロードされるべきアプリケーションに同一の applicationID を付与するよう記述されたデータ管理テーブルの一例を示す図である。

図48 (b) は、メモリ規模が小さい再生装置におけるローカルメモリ29の格納内容の変遷を示す図である。

図48 (c) は、メモリ規模が大きい再生装置におけるローカルメモリ29の格納内容の変遷を示す図である。

図49 は、データ管理テーブルに基づくアプリケーションマネージャ36によるロード処理の処理手順を示す図である。

図50 は、アプリケーションqの生存区間に、現在の再生時点が到達した場合のアプリケーションマネージャ36による処理手順を示す図である。

図51 は、Java 仮想マシン38によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

図52 (a) は、第7実施形態に係る BD-J オブジェクトの内部構成を示す図である。

図52 (b) は、プレイリスト管理テーブルの一例を示す図である。

図52 (c) は、分岐先タイトルのプレイリスト管理テーブルにおいて、再生属性が AutoPlay に設定された PL が存在する場合、再生装置がどのような処理を行うかを示す図である。

図53 (a) は、再生属性が非自動再生を示すよう設定された場合の非 AV 系タイトルにおけるタイトル時間軸を示す図である。

図53 (b) は、再生属性が AutoPlay に設定された非 AV 系タイトルのタイトル時間軸を示す図である。

- 5 図53 (c) は、プレイリスト管理テーブルにおいて再生属性が” AutoPlay” を示すよう設定され、アプリケーションが強制終了した場合を示す図である。

図53 (d) は、プレイリスト管理テーブルにおいて再生属性が” AutoPlay” を示すよう設定され、メインアプリの起動に失敗したケースを示す図である。

- 10 図54 は、第7実施形態に係るアプリケーションマネージャ36の処理手順を示すフローチャートである。

図55 は、プレイリスト管理テーブルにおいて” 再生属性=AutoPlay” に設定されることにより、どのような再生が行われるかを模式化した図である。

図56 (a) (b) は、アプリケーションの扱いと、起動属性との関係を示した図である。

- 15 図57 は、第8実施形態に係る Java 仮想マシン38によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

図58 (a) (b) は、第9実施形態に係る読込優先度の一例を示す図である。

図59 (a) は、グループ属性が付与されたデータ管理テーブルを示す図である。

- 20 図59 (b) は、アプリケーション管理テーブルに基づくローカルメモリ29に対するアクセスを示す図である。

図60 は、アプリケーション管理テーブルの割当単位のバリエーションを示す図である。

- 25 発明を実施するための最良の形態
(第1実施形態)

以降、本発明に係る再生装置の実施形態について説明する。先ず始めに、本発明に係る再生装置の実施行為のうち、使用行為についての形態を説明する。

図1 は、本発明に係る再生装置の、使用行為についての形態を示す図である。

図1において、本発明に係る再生装置は再生装置200であり、テレビ300、リモコン400と共にホームシアターシステムを形成する。

このBD-ROM100は、再生装置200、リモコン300、テレビ400により形成されるホームシアターシステムに、映画作品を供給するという用途
5 に供される。

以上が本発明に係る再生装置の使用形態についての説明である。

続いて本発明に係る再生装置の再生の対象となる、記録媒体であるBD-ROMについて説明する。BD-ROMにより、ホームシアターシステムに供給されるディスクコンテンツは、互いに分岐可能な複数タイトルから構成される。各タイトルは、1つ以上のプレイリストと、このプレイリストを用いた動的な制御
10 手順とからなる。

プレイリストとは、1つ以上のデジタルストリームと、そのデジタルストリームにおける再生経路とから構成され、“時間軸”の概念をもつBD-ROM上のアクセス単位である。以上のプレイリストと、動的な制御手順とを包含しているため、タイトルはデジタルストリーム特有の時間軸の概念と、コンピュータ
15 プログラム的な性質とを併せもっている。

図2は、BD-ROMにおけるファイル・ディレクトリ構成を示す図である。本図においてBD-ROMには、Rootディレクトリの下に、BDMVディレクトリがある。

20 BDMVディレクトリには、拡張子bdmvが付与されたファイル(index.bdmv,MovieObject.bdmv)と、拡張子BD-Jが付与されたファイル(00001.BD-J,00002.BD-J,00003.BD-J)がある。そしてこのBDMVディレクトリの配下には、更にPLAYLISTディレクトリ、CLIPINFディレクトリ、STREAMディレクトリ、BDARディレクトリと呼ばれる4つのサブディレ
25 クトリが存在する。

PLAYLISTディレクトリには、拡張子mplsが付与されたファイル(00001.mpls,00002.mpls,00003.mpls)がある。

CLIPINFディレクトリには、拡張子clpiが付与されたファイル(00001.clpi,00002.clpi,00003.clpi)がある。

STREAM ディレクトリには、拡張子 m2ts が付与されたファイル
(00001.m2ts,00002.m2ts,00003.m2ts)がある。

BDAR ディレクトリには、拡張子 jar が付与されたファイル
(00001.jar,00002.jar,00003.jar)がある。以上のディレクトリ構造により、互い
5 に異なる種別の複数ファイルが、BD-ROM 上に配置されていることがわかる。

本図において拡張子.m2ts が付与されたファイル
(00001.m2ts,00002.m2ts,00003.m2ts……)は、AVClip を格納している。
AVClip には、MainClip、SubClip といった種別がある。MainClip は、ビデ
オストリーム、オーディオストリーム、プレゼンテーショングラフィクススト
10 リーム、インタラクティブグラフィクスストリームというような複数エレメン
タリストリームを多重化することで得られたデジタルストリームである。

SubClip は、オーディオストリーム、グラフィクスストリーム、テキスト字
幕ストリーム等、1 つのエレメンタリストリームのみにあたるデジタルストリ
ームである。

15 拡張子” clpi” が付与されたファイル(00001.clpi,00002.clpi,00003.clpi……)
は、AVClip のそれぞれに 1 対 1 に対応する管理情報である。管理情報故に、
Clip 情報は、AVClip におけるストリームの符号化形式、フレームレート、ピ
ットレート、解像度等の情報や、頭出し位置を示す EP_map をもっている。

拡張子” mpls” が付与されたファイル
20 (00001.mpls,00002.mpls,00003.mpls……)は、プレイリスト情報を格納した
ファイルである。プレイリスト情報は、AVClip を参照してプレイリストを定
義する情報である。プレイリストは、MainPath 情報、PLMark 情報、SubPath
情報から構成される。

MainPath 情報は、複数の PlayItem 情報からなる。PlayItem とは、1 つ以
25 上の AVClip 時間軸上において、In_Time,Out_Time を指定することで定義さ
れる再生区間である。PlayItem 情報を複数配置させることで、複数再生区間
からなるプレイリスト(PL)が定義される。図 3 は、AVClip と、PL との関係を示す図である。第 1 段目は AVClip がもつ時間軸を示し、第 2 段目は、PL がも
つ時間軸を示す。PL 情報は、PlayItem#1,#2,#3 という 3 つの PlayItem 情報

を含んでおり、これら PlayItem#1,#2,#3 の In_time,Out_time により、3つの再生区間が定義されることになる。これらの再生区間を配列させると、AVClip 時間軸とは異なる時間軸が定義されることになる。これが第 2 段目に示す PL 時間軸である。このように、PlayItem 情報の定義により、AVClip とは異なる
5 時間軸の定義が可能になる。

AVClip に対する指定は、原則 1 つであるが、複数 AVClip に対する一括指定もあり得る。この一括指定は、PlayItem 情報における複数の Clip_Information_file_name によりなされる。図 4 は、4 つの Clip_Information_file_name によりなされた一括指定を示す図である。本図に
10 において第 1 段目～第 4 段目は、4 つの AVClip 時間軸(AVClip#1,#2,#3,#4 の時間軸)を示し、第 5 段目は、PL 時間軸を示す。PlayItem 情報が有する、4 つの Clip_Information_file_name にて、これら 4 つの時間軸が指定されている。こうすることで、PlayItem が有する In_time,Out_time により、択一的に再生可能な 4 つの再生区間が定義されることになる。これにより、PL 時間軸には、
15 切り換え可能な複数アングル映像からなる区間(いわゆるマルチアングル区間)が定義されることになる。

PLmark 情報は、PL 時間軸のうち、任意の区間を、チャプターとして指定する情報である。図 5 は、PLmark によるチャプター定義を示す図である。本図において第 1 段目は、AVClip 時間軸を示し、第 2 段目は PL 時間軸を示す。
20 図中の矢印 pk1,2 は、PLmark における PlayItem 指定(ref_to_PlayItem_Id)と、一時点の指定(mark_time_stamp)とを示す。これらの指定により PL 時間軸には、3 つのチャプター(Chapter#1,#2,#3)が定義されることになる。

SubPath 情報は、複数の SubPlayItem 情報からなる。SubPlayItem 情報は、SubClip の時間軸上に In_Time,Out_Time を指定することで再生区間を定義する。
25 る。また SubPlayItem 情報は、SubClip 時間軸上の再生区間を、PL 時間軸に同期させるという同期指定が可能であり、この同期指定により、PL 時間軸と、SubPlayItem 情報時間軸とは同期して進行することになる。図 6 は、SubPlayItem 時間軸上の再生区間定義と、同期指定を示す図である。本図において第 1 段目は、PL 時間軸を示し、第 2 段目は SubPlayItem 時間軸を示す。

図中の SubPlayItem.IN_time は再生区間の始点を、SubPlayItem.Out_time は再生区間の終点をそれぞれ示す。これにより SubClip 時間軸上にも再生区間が定義されていることがわかる。矢印 Sn1 において Sync_PlayItem_Id は、PlayItem に対する同期指定を示し、矢印 Sn2 において

- 5 sync_start_PTS_of_PlayItem は、PL 時間軸における PlayItem 上の一時点の指定を示す。

- 複数 AVClip の切り換えを可能とするマルチアングル区間や、AVClip - SubClip を同期させ得る同期区間の定義を可能とするのが、BD-ROM におけるプレイリスト情報の特徴である。以上の Clip 情報及びプレイリスト情報は、
- 10 静的シナリオ”に分類される。何故なら、以上の Clip 情報及びプレイリスト情報により、静的な再生単位である PL が定義されるからである。以上で静的シナリオについての説明を終わる。

- 続いて”動的なシナリオ”について説明する。動的シナリオとは、AVClip の再生制御を動的に規定するシナリオデータである。”動的に”というのは、再生装置における状態変化やユーザからのキーイベントにより再生制御の中身がかわることをいう。BD-ROM では、この再生制御の動作環境として 2 つのモードを想定している。1 つ目は、DVD 再生装置の動作環境と良く似た動作環境であり、コマンドベースの実行環境である。2 つ目は、Java 仮想マシンの動作環境である。これら 2 つの動作環境のうち 1 つ目は、HDMV モードと呼ばれ
- 20 る。2 つ目は、BD-J モードと呼ばれる。これら 2 つの動作環境があるため、動的シナリオはこのどちらかの動作環境を想定して記述される。HDMV モードを想定した動的シナリオは Movie オブジェクトと呼ばれ、管理情報により定義される。一方 BD-J モードを想定した動的シナリオは BD-J オブジェクトと呼ばれる。

- 25 先ず初めに Movie オブジェクトについて説明する。

<Movie オブジェクト> .

Movie オブジェクトは、”タイトル”の構成要素であり、ファイル MovieObject.bdmv に格納される。図 7 (a) は、Movie オブジェクトの内部構成を示す図である。Movie オブジェクトは、属性情報、複数のナビゲーション

ンコマンドからなるコマンド列からなる。

- 属性情報は、PL 時間軸において、MenuCall がなされた際、MenuCall 後の再生再開を意図しているか否かを示す情報(resume_intention_flag)、PL 時間軸において MenuCall をマスクするかを示す情報(menu_call_mask)、タイトル
- 5 サーチをマスクするかを示す情報(title_search_flag)からなる。Movie オブジェクトは、“時間軸” + “プログラムの制御” という 2 つの性質を併せ持つことができるで、本編再生を実行するもの等、多様な種類のタイトルがこの Movie オブジェクトにより記述されることになる。

- ナビゲーションコマンド列は、条件分岐、再生装置における状態レジスタの設定、状態レジスタの設定値取得等を実現するコマンド列からなる。Movie オブジェクトにおいて記述可能なコマンドを以下に示す。
- 10

PlayPL コマンド

書式: PlayPL(第 1 引数, 第 2 引数)

- 15 第 1 引数は、プレイリストの番号で、再生すべき PL を指定することができる。第 2 引数は、その PL に含まれる PlayItem や、その PL における任意の時刻、Chapter、Mark を用いて再生開始位置を指定することができる。

PlayItem により PL 時間軸上の再生開始位置を指定した PlayPL 関数を PlayPLatPlayItem()、

- 20 Chapter により PL 時間軸上の再生開始位置を指定した PlayPL 関数を PlayPLatChapter()、

時刻情報により PL 時間軸上の再生開始位置を指定した PlayPL 関数を PlayPLatSpecified Time()という。

25 JMP コマンド

書式: JMP 引数

JMP コマンドは、現在の動的シナリオを途中で廃棄し(discard)、引数たる分岐先動的シナリオを実行するという分岐である。JMP 命令の形式には、分岐先動的シナリオを直接指定している直接参照のものと、分岐先動的シナリオを間

接参照している間接参照のものがある。

5 Movie オブジェクトにおけるナビゲーションコマンドの記述は、DVD におけるナビゲーションコマンドの記述方式と良く似ているので、DVD 上のディスクコンテンツを、BD-ROM に移植するという作業を効率的に行うことができる。Movie オブジェクトについては、以下の国際公開公報に記載された先行技術が存在する。詳細については、本国際公開公報を参照されたい。

10 国際公開公報 WO 2004/074976

以上で Movie オブジェクトについての説明を終える。続いて BD-J オブジェクトについて説明する。

<BD-J オブジェクト>

15 拡張子 BD-J が付与されたファイル(00001.BD-J,00002.BD-J,00003.BD-J)は、BD-J オブジェクトを構成する。BD-J オブジェクトは、Java プログラミング環境で記述された、BD-J モードの動的シナリオである。図7 (b) は、BD-J オブジェクトの内部構成を示す図である。本図に示すように BD-J オブジェクトは、Movie オブジェクト同様の属性情報、アプリケーション管理テーブルからなる。属性情報を有している点で BD-J オブジェクトは Movie オブジェクトとほぼ同じである。Movie オブジェクトとの違いは、BD-J オブジェクトはコマンドが直接記述されていない点である。つまり Movie オブジェクトにおいて制御手順は、ナビゲーションコマンドにより直接記述されていた。これ
20 に対し BD-J オブジェクトでは、そのタイトルを生存区間としている Java アプリケーションをアプリケーション管理テーブル上に定めることにより、間接的に制御手順を規定している。このような間接的な規定により、複数タイトルにおいて制御手順を共通化するという、制御手順の共通化を効率的に行うことができる。

図7 (c) は、Java アプリケーションの内部構成を示す図である。本図にお

いてアプリケーションは、仮想マシンのヒープ領域(ワークメモリとも呼ばれる)にロードされた 1 つ以上の xlet プログラムからなる。このワークメモリでは、1 つ以上のスレッドが動作しており、ワークメモリにロードされた xlet プログラム、及び、スレッドから、アプリケーションは構成されることになる。

5 以上がアプリケーションの構成である。

このアプリケーションの実体にあたるのが、BDMV ディレクトリ配下の BDAR ディレクトリに格納された Java アーカイブファイル (00001.jar,00002.jar)である。以降、Java アーカイブファイルについて説明する。

10 Java アーカイブファイル(00001.jar,00002.jar)は、Java アプリケーションを構成するプログラム、データを格納したアーカイブファイルである。図 8 (a)は、アーカイブファイルにより収められているプログラム、データを示す図である。本図におけるデータは、枠内に示すディレクトリ構造が配置された複数
15 ファイルを、java アーカイバでまとめたものである。枠内に示すディレクトリ構造は、root ディレクトリ、java ディレクトリ、image ディレクトリとからなり、root ディに common.pkg が、java ディレクトリに aaa.class,bbb.class が、image ディレクトリに、menu.jpg が配置されている。java アーカイブファイルは、これらを java アーカイバでまとめることで得られる。かかるデータは、BD-ROM からキャッシュに読み出されるにあたって展開され、キャッシュ上
20 で、ディレクトリに配置された複数ファイルとして取り扱われる。Java アーカイブファイルのファイル名における"xxxxxx"という 5 桁の数値は、アプリケーションの ID(applicationID)を示す。本 Java アーカイブファイルがキャッシュに読み出された際、このファイル名における数値を参照することにより、任意の Java アプリケーションを構成するプログラム、データを取り出すことが
25 できる。

Java アーカイブファイルにおいて 1 つにまとめられるファイルには、xlet プログラムがある。

xlet プログラムは、JMF(Java Media FrameWork)インターフェイスを利用することができる Java プログラムである。xlet プログラムは、キーイベント

を受信する EventListner 等、複数の関数からなり、JMF 等の方式に従って、受信したキーイベントに基づく処理を行う。

図 8 (b) は、xlet プログラムの一例を示す図である。JMF A"BD://00001.mpls" は、PL を再生するプレーヤインスタンスの生成を Java 仮想マシンに命じるメソッドである。A.play は、JMF プレーヤインスタンスに再生を命じるメソッドである。かかる JMF プレーヤインスタンス生成は、JMF ライブラリに基づきなされる。xlet プログラムの記述は、BD-ROM の PL に限らず、時間軸をもったコンテンツ全般に適用可能な JMF の記述である。このような記述が可能であるので、Java プログラミングに長けたソフトハウスに、BD-J オブジェクト作成を促すことができる。

図 8 (b) における JumpTitle0; は、ファンクション API のコールである。このファンクション API は、他のタイトルへの分岐(図中では title#1)を再生装置に命じるものである。ここでファンクション API とは、BD-ROM 再生装置により供給される API(Application Interface)である。JumpTitle コマンドの他にも、ファンクション API のコールにより、BD-ROM 再生装置特有の処理を xlet プログラムに記述することができる。

BD-J モードにおいて PL 再生は、JMF インターフェイスにより規定される。この JMF プレーヤインスタンスは、PL 時間軸を定めるものだから、タイトル時間軸は、この JMF プレーヤインスタンスをもったタイトルから定まる。また

BD-J モードにおいてタイトルからタイトルへの分岐は JumpTitleAPI のコールにより規定される。JumpTitleAPI コールは、いわばタイトルの終了時点を定めるものなので、こうした JMF プレーヤインスタンス、JumpTitleAPI コールをもったアプリケーションが、BD-J モードにおいてタイトルの開始及び終了を律することになる。かかるアプリケーションを本編再生アプリケーションという。

以上が、BD-J モードにおける動的シナリオについての説明である。この BD-J モードにおける動的シナリオにより、PL 再生と、プログラムの制御とを併せもったタイトルが定義されることになる。尚、本実施形態においてアプリ

ケーションを構成するプログラム、データは、Java アーカイブファイルにまとめられたが、LZH ファイル、zip ファイルであってもよい。

<タイトル時間軸>

- タイトルを構成する静的シナリオ、動的シナリオについて説明を終えたところで、これらによりどのような時間軸が定義されるかについて説明する。タイトルにより定義される時間軸は、“タイトル時間軸”と呼ばれる。タイトル時間軸とは、Movie オブジェクト、又は、BD-J オブジェクトにより再生が命じられる PL により構成される。ここで一例を挙げるのは、図9 (a) のようなタイトルである。このタイトルは、トップメニュー→title#1→title#2→トップメニュー、トップメニュー→title#3→トップメニューという一連のタイトルである。かかるタイトルのうち、title#1 は PlayList#1、PlayList#2、title#2 が PlayList#3、title#3 が PlayList#4 の再生を命じるものなら、図9 (b) のように、PlayList#1、PlayList#2 の時間軸を足し合わせた時間軸を、title#1 はもつことになる。同様に title#2 は、PlayList#3 時間軸からなる時間軸を、title#3 は PlayList#4 時間軸からなる時間軸を持つことになる。これらタイトル時間軸における PL 時間軸ではシームレス再生が保証されるが、タイトル時間軸間ではシームレス再生の保証は必要でなくなる。Java アプリケーションを動作させるにあたっては、Java アプリケーションを、仮想マシンのワークメモリ上に存在させてもよい期間(サービス期間)を、こうしたタイトル時間軸上に定義せねばならない。BD-J モードにおいて Java アプリケーションを動作させるにあたっては、互いに分岐し合う時間軸上に、Java アプリケーションのサービス期間を定義せねばならない。このサービス期間の定義が、BD-ROM 向けのプログラミングを行うにあたっての留意点になる。

- 最後に、index.bdmv に格納された IndexTable について説明する。
- IndexTable は、タイトル番号と、Movie オブジェクト、BD-J オブジェクトとを対応づけるテーブルであり、動的シナリオから動的シナリオへの分岐の際、参照される間接参照用テーブルである。IndexTable は、複数ラベルのそれぞれに対する Index からなる。各 Index には、そのラベルに対応する動的シナリオの識別子が記述されている。こうした IndexTable を参照することで、Movie

オブジェクト、BD-J オブジェクトの違いを厳密に区別することなく、分岐を実現することができる。IndexTable については、以下の国際公開公報に詳細が記載されている。詳細については、本公報を参照されたい。

5 国際公開公報 WO 2004/025651 A1 公報

以上が BD-ROM に記録されているファイルについて説明である。

<アプリケーション管理テーブル>

10 JMF プレーヤインスタンス、JumpTitleAPI コールをもったアプリケーションが、タイトル時間軸を律することは上述した通りだが、JMF プレーヤインスタンスや JumpTitleAPI のコールをもたないその他のアプリケーションを、タイトル時間軸上で動作させる場合、時間軸の何処からアプリケーションによるサービスを開始し、時間軸の何処でアプリケーションによるサービスを終えるかという”サービスの開始点・終了点”を明確に規定することが重要になる。本
15 実施形態では、アプリケーションによるサービスが開始してから、終了するまでを、”アプリケーションの生存”として定義する。アプリケーションの生存を定義するための情報は、BD-J オブジェクトにおけるアプリケーション管理テーブルに存在する。以降アプリケーション管理テーブルについてより詳しく説明する。

20 アプリケーション管理テーブル(AMT)は、各タイトルが有しているタイトル時間軸において、仮想マシンのワークメモリ上で生存し得るアプリケーションを示す情報である。ワークメモリにおける生存とは、そのアプリケーションを構成する xlet プログラムが、ワークメモリに読み出され、仮想マシンによる実行が可能になっている状態をいう。図 7 (b) における破線矢印 at1 は、ア
25 プリケーション管理テーブルの内部構成をクローズアップして示す。この内部構成に示すように、アプリケーション管理テーブルは、『生存区間』と、そのタイトルを生存区間としているアプリケーションを示す『applicationID』と、そのアプリケーションの『起動属性』とからなる。

近い将来、実施されるであろうディスクコンテンツを題材に選んで、アプリ

ケーション管理テーブルにおける生存区間記述について、具体例を交えて説明する。ここで題材にするディスクコンテンツは、映像本編を構成する本編タイトル(title#1)、オンラインショッピングを構成するオンラインショッピングタイトル(title#2)、ゲームアプリケーションを構成するゲームタイトル(title#3)という、性格が異なる3つのタイトルを含むものである。図10は、本編タイトル、オンラインショッピングタイトル、ゲームタイトルという3つのタイトルを含むディスクコンテンツを示す図である。本図における右側には IndexTable を記述しており、左側には3つのタイトルを記述している。

右側における破線枠は、各アプリケーションがどのタイトルに属しているかという帰属関係を示す。3つのタイトルのうち title#1 は、application#1、application#2、application#3 という3つのアプリケーションからなる。title#2 は、application#3、application#4 という2つのアプリケーション、title#3 は、application#5 を含む。図11は、図10に示した3つのタイトルの再生画像の一例を示す図である。これら3つのタイトルの再生画像において、図11(a) (b) の本編タイトル、オンラインショッピングタイトルには、ショッピングカートを描した映像(カート cr1)1が存在するが、図11(c)のゲームタイトルには、カート映像が存在しない。カート cr1 は、本編タイトル、オンラインショッピングタイトルにおいて共通して表示しておく必要があるので、カートプログラムたる application#3 を、title#1、title#2 の双方で起動するようにしている。このように複数タイトルで起動するようなアプリケーションには、上述したカートアプリの他に、映画作品に登場するマスコットを描したエージェントアプリ、メニューコール操作に応じてメニュー表示を行うメニューアプリがある。

図10の破線に示される帰属関係から各アプリケーションの生存区間をグラフ化すると、図12(a)のようになる。本図において横軸は、タイトル時間軸であり、縦軸方向に各アプリケーションの生存区間を配置している。ここで application#1、application#2 は、title#1 のみに帰属しているので、これらの生存区間は、title#1 内に留まっている。application#4 は、title#2 のみに帰属しているので、これらの生存区間は、title#2 内に留まっている。application#5

は、title#3 のみに帰属しているので、これらの生存区間は、title#3 内に留まっている。application#3 は、title#1 及び title#2 に帰属しているので、これらの生存区間は、title#1-title#2 にわたる。この生存区間に基づき、アプリケーション管理テーブルを記述すると、title#1,#2,#2 のアプリケーション管理テーブルは図 1 2 (b) のようになる。このようにアプリケーション管理テーブルが記述されれば、title#1 の再生開始時において application#1、application#2、application#3 をワークメモリにロードしておく。そして title#2 の開始時に application#1、application#2 をワークメモリから削除して application#3 のみにするという制御を行う。これと同様に title#2 の再生開始時において application#4 をワークメモリにロードしておき、title#3 の開始時に application#3,#4 をワークメモリから削除するという制御を行いうる。

更に、title#3 の再生中において application#5 をワークメモリにロードしておき、title#3 の再生終了時に application#5 をワークメモリから削除するという制御を行いうる。

タイトル間分岐があった場合でも、分岐元一分岐先において生存しているアプリケーションはワークメモリ上に格納しておき、分岐元にはなく、分岐先にのみ存在するアプリケーションをワークメモリに読み込めば良いから、アプリケーションをワークメモリに読み込む回数は必要最低数になる。このように、読込回数を少なくすることにより、タイトルの境界を意識させないアプリケーション、つまりアンバウンダリなアプリケーションを実現することができる。

続いてアプリケーションの起動属性について説明する。起動属性には、自動的な起動を示す「AutoRun」、自動起動の対象ではないが、仮想マシンのワークメモリに置いて良いことを示す「Persistent」、仮想マシンのワークメモリにはおかれるが、CPU パワーの割り当ては不可となる「Suspend」がある。

「AutoRun」は、対応するタイトルの分岐と同時に、そのアプリケーションをワークメモリに読み込み、且つ実行する旨を示す生存区間である。あるタイトルから、別のタイトルへの分岐があると、アプリケーション管理を行う管理主体(アプリケーションマネージャ)は、その分岐先タイトルにおいて生存しており、かつ起動属性が AutoRun に設定されたアプリケーションを仮想マシン

のワークメモリに読み込み実行する。これによりそのアプリケーションは、タイトル分岐と共に自動的に起動されることになる。起動属性を **AutoRun** に設定しておくアプリケーションとしては、**JMF** プレーヤインスタンス及び **JumpTitleAPI** コールをもつようなアプリケーションが挙げられる。何故なら、
5 このようなアプリケーションは、タイトル時間軸を律する側のアプリケーションであり、このようなアプリケーションを自動的に起動にしないと、タイトル時間軸の概念が曖昧になってしまうからである。

起動属性「**Persistent**」は、継続属性であり、分岐元 title におけるアプリケーションの状態を継続することを示す。またワークメモリにロードしてよいことを示す属性である。起動属性が「**Persistent**」である場合、この起動属性が
10 付与されたアプリケーションは、他のアプリケーションからの呼び出しが許可されることになる。アプリケーション管理を行う管理主体(アプリケーションマネージャ)は、起動中のアプリケーションから呼出があると、そのアプリケーションの **applicationID** が、アプリケーション管理テーブルに記述されていて、
15 起動属性が「**Persistent**」であるか否かを判定する。「**Persistent**」であれば、そのアプリケーションをワークメモリにロードする。一方、その呼出先アプリケーションの **applicationID** がアプリケーション管理テーブルに記述されていない場合、そのアプリケーションはワークメモリにロードされない。アプリケーションによる呼出は、この「**Persistent**」が付与されたアプリケーションに限られることになる。
20

「**Persistent**」は、起動属性を明示的に指定しない場合に付与されるデフォルトの起動属性であるから、あるアプリケーションの起動属性が無指定「—」である場合、そのアプリケーションの起動属性の起動属性はこの **Persistent** であることを意味する。

25 これらの起動属性が、図 11 のアプリケーションにおいてどのように記述されているかについて説明する。図 13 は、図 12 の 3 つのアプリケーションに対する起動属性の設定例である。図 12 に示した 3 つのアプリケーションのうち **application#2** は、図 13 (b) に示すように他のアプリケーションからのアプリケーション呼出があって初めて起動するアプリケーションであるとする。

残りの application#1、application#3 は、title#1 の開始と同時に自動的に起動されるアプリケーションであるとする。この場合、図 1 3 (a) に示すように、アプリケーション管理テーブルにおける各アプリケーションの起動属性を、application#1、application#3 は「AutoRun」、application#2 は、「Persistent」
5 と設定しておく。この場合、application#1、application#3 は、title#1 への分岐時において自動的にワークメモリにロードされ、実行されることになる。一方 application#2 は、起動属性が Persistent なので、「application#3 は仮想マシンのワークメモリ上にロードしてよいアプリケーション」であるとの消極的な意味に解される。故に、application#2 は、application#1 からの呼出があっ
10 て初めて仮想マシンのワークメモリにロードされ、実行されることになる。以上の生存区間・起動属性により、仮想マシン上で動作し得るアプリケーションの数を 4 個以下に制限し、総スレッド数を 64 個以下に制限することが可能なので、アプリケーションの安定動作を保証することができる。

続いて Suspend について説明する。

15 Suspend とは、リソースは割り付けられているが、CPU パワーは割り当てられない状態にアプリケーションが置かれることをいう。かかる Suspend は、例えばゲームタイトルの実行中に、サイドパスを経由するという処理の実現に有意義である。図 1 4 (a) (b) は Suspend が有意義となる事例を示す図である。図 1 4 (b) に示すように、3 つのタイトル(title#1、title#2、title#3)
20 があり、そのうち title#1、title#3 はゲームアプリを実行するが、途中の title#2 はサイドパスであり、映像再生を実現するものである。サイドパスでは、映像再生を実現する必要があるため、ゲームの実行を中断させることになる。ゲームアプリでは途中のスコア等が計数されているため、リソースの格納値は title#2 の前後で維持したい。この場合、title#2 の開始時点でゲームアプリを
25 Suspend し、title#3 の開始時点で application#2 をレジュームするというようにアプリケーション管理テーブルを記述する。こうすることで title#2 において application#2 は、リソースは割り付けられているので、リソースの格納値は維持される。しかし、CPU パワーは割り当てられない状態なので仮想マシンにより application#2 は実行されることはない。これにより、ゲームタイトル

の実行中に、サイドパスを実行するという処理が実現される。

図15は、起動属性がとり得る三態様(Persistent、AutoRun、Suspend)と、直前タイトルにおけるアプリケーション状態の三態様(非起動、起動中、Suspend)とがとりうる組合せを示す図である。直前状態が”非起動”である場合、起動属性が”AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、起動されることになる。

直前状態が”非起動”であり、起動属性が”Persistent”、“Suspend”であるなら、分岐先タイトルにおいてそのアプリケーションは、何もせず、状態を継続することになる。

10 直前状態が”起動中”である場合、起動属性が”Persistent”、“AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、何もせず、状態を継続することになる。

起動属性が”Suspend”であるなら、アプリケーションの状態は Suspend されることになる。直前状態が”Suspend”である場合、分岐先タイトルの起動属性が”Suspend”なら Suspend を維持することになる。”Persistent”、“AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、レジュームすることになる。アプリケーション管理テーブルにおいて生存区間及び起動属性を定義することにより、タイトル時間軸の進行に沿って、Java アプリケーションを動作させるという同期制御が可能になり、映像再生と、プログラム実行とを伴った、様々なアプリケーションを世に送り出すことができる。以上が記録媒体についての説明である。続いて本発明に係る再生装置について説明する。

図16は、本発明に係る再生装置の内部構成を示す図である。本発明に係る再生装置は、本図に示す内部に基づき、工業的に生産される。本発明に係る再生装置は、主としてシステム LSI と、ドライブ装置という2つのパーツからなり、これらのパーツを装置のキャビネット及び基板に実装することで工業的に生産することができる。システム LSI は、再生装置の機能を果たす様々な処理部を集積した集積回路である。こうして生産される再生装置は、BD-ROM ドライブ1、リードバッファ2、デマルチプレクサ3、ビデオデコーダ4、ビデ

5 オブレーション 5、P-Graphics デコーダ 9、Presentation Graphics プレーン 10、
合成部 11、フォントゼネレータ 12、I-Graphics デコーダ 13、スイッチ 1
4、Interactive Graphics プレーン 15、合成部 16、HDD 17、リードバッ
ファ 18、デマルチプレクサ 19、オーディオデコーダ 20、シナリオメモリ
21、CPU 22、キーイベント処理部 23、命令 ROM 24、スイッチ 25、
CLUT 部 26、CLUT 部 27、PSR セット 28、ローカルメモリ 29 から構
成される。

BD-ROM ドライブ 1 は、BD-ROM のローディング/イジェクトを行い、
BD-ROM に対するアクセスを実行する。

10 リードバッファ 2 は、FIFO メモリであり、BD-ROM から読み出された TS
パケットが先入れ先出し式に格納される。

デマルチプレクサ(De-MUX) 3 は、リードバッファ 2 から TS パケットを取
り出して、この TS パケットを構成する TS パケットを PES パケットに変換す
る。そして変換により得られた PES パケットのうち、CPU 22 から設定され
15 た PID をもつものをビデオデコーダ 4、オーディオデコーダ 20、P-Graphics
デコーダ 9、I-Graphics デコーダ 13 のどれかに出力する。

ビデオデコーダ 4 は、デマルチプレクサ 3 から出力された複数 PES パケッ
トを復号して非圧縮形式のピクチャを得てビデオプレーン 5 に書き込む。

ビデオプレーン 5 は、非圧縮形式のピクチャを格納しておくためのプレーン
20 である。プレーンとは、再生装置において一画面分の画素データを格納してお
くためのメモリ領域である。再生装置に複数のプレーンを設けておき、これら
プレーンの格納内容を画素毎に加算して、映像出力を行えば、複数の映像内容
を合成させた上で映像出力を行うことができる。ビデオプレーン 5 における解
像度は 1920×1080 であり、このビデオプレーン 5 に格納されたピクチャデー
25 タは、16 ビットの YUV 値で表現された画素データにより構成される。

P-Graphics デコーダ 9 は、BD-ROM、HDD 17 から読み出されたプレゼン
テーショングラフィクスストリームをデコードして、非圧縮グラフィクスを
Presentation Graphics プレーン 10 に書き込む。グラフィクスストリームの
デコードにより、字幕が画面上に現れることになる。

Presentation Graphics プレーン 10 は、一画面分の領域をもったメモリであり、一画面分の非圧縮グラフィクスを格納することができる。本プレーンにおける解像度は 1920×1080 であり、Presentation Graphics プレーン 10 中の非圧縮グラフィクスの各画素は 8 ビットのインデックスカラーで表現される。

- 5 CLUT(Color Lookup Table)を用いてかかるインデックスカラーを変換することにより、Presentation Graphics プレーン 10 に格納された非圧縮グラフィクスは、表示に供される。

合成部 11 は、非圧縮状態のピクチャデータ(i)を、Presentation Graphics プレーン 10 の格納内容と合成する。

- 10 フォントゼネレータ 12 は、文字フォントを用いて textST ストリームに含まれるテキストコードをビットマップに展開する。

I-Graphics デコーダ 13 は、BD-ROM 又は HDD 17 から読み出されたインタラクティブグラフィクスストリームをデコードして、非圧縮グラフィクスを Interactive Graphics プレーン 15 に書き込む。

- 15 スイッチ 14 は、フォントゼネレータ 12 が生成したフォント列、P-Graphics デコーダ 9 のデコードにより得られたグラフィクスの何れかを選択的に Presentation Graphics プレーン 10 に書き込むスイッチである。

Interactive Graphics プレーン 15 は、I-Graphics デコーダ 13 によるデコードで得られた非圧縮グラフィクスが書き込まれる。

- 20 合成部 16 は、Interactive Graphics プレーン 10 の格納内容と、合成部 8 の出力である合成画像(非圧縮状態のピクチャデータと、Presentation Graphics プレーン 7 の格納内容とを合成したもの)とを合成する。

- HDD 17 は、ネットワーク等を介してダウンロードされた SubClip、Clip 情報、プレイリスト情報が格納される内蔵媒体である。この HDD 17 中のプレイリスト情報は BD-ROM 及び HDD 17 のどちらに存在する Clip 情報であっても、指定できる点で異なる。この指定にあたって、HDD 17 上のプレイリスト情報は、BD-ROM 上のファイルをフルパスで指定する必要はない。本 HDD 17 は、BD-ROM と一体になって、仮想的な 1 つのドライブ(バーチャルパッケージと呼ばれる)として、再生装置により認識されるからである。故に、
- 25

PlayItem 情報における Clip_Information_file_name 及び SubPlayItem 情報の Clip_Information_file_name は、Clip 情報の格納したファイルのファイルボディにあたる 5 桁の数値を指定することにより、HDD 17、BD-ROM 上の AVClip を指定することができる。この HDD の記録内容を読み出し、BD-ROM の記録内容と動的に組み合わせることにより、様々な再生のバリエーションを産み出すことができる。

リードバッファ 18 は、FIFO メモリであり、HDD 17 から読み出された TS パケットが先入れ先出し式に格納される。

デマルチプレクサ(De-MUX) 19 は、リードバッファ 18 から TS パケットを取り出して、TS パケットを PES パケットに変換する。そして変換により得られた PES パケットのうち、所望の streamPID をもつものをフロントゼネレータ 12 に出力する。

オーディオデコーダ 20 は、デマルチプレクサ 19 から出力された PES パケットを復号して、非圧縮形式のオーディオデータを出力する。

シナリオメモリ 21 は、カレントの PL 情報やカレントの Clip 情報を格納しておくためのメモリである。カレント PL 情報とは、BD-ROM に記録されている複数 PL 情報のうち、現在処理対象になっているものをいう。カレント Clip 情報とは、BD-ROM に記録されている複数 Clip 情報のうち、現在処理対象になっているものをいう。

CPU 22 は、命令 ROM 24 に格納されているソフトウェアを実行して、再生装置全体の制御を実行する。

キーイベント処理部 23 は、リモコンや再生装置のフロントパネルに対するキー操作に応じて、その操作を行うキーイベントを出力する。

命令 ROM 24 は、再生装置の制御を規定するソフトウェアを記憶している。

スイッチ 25 は、BD-ROM 及び HDD 17 から読み出された各種データを、リードバッファ 2、リードバッファ 18、シナリオメモリ 21、ローカルメモリ 29 のどれかに選択的に投入するスイッチである。

CLUT 部 26 は、ビデオプレーン 5 に格納された非圧縮グラフィクスにおけるインデックスカラーを、Y,Cr,Cb 値に変換する。

CLUT 部 2 7 は、Interactive Graphics プレーン 1 5 に格納された非圧縮グラフィックスにおけるインデックスカラーを、Y,Cr,Cb 値に変換する。

PSR セット 2 8 は、再生装置に内蔵されるレジスタであり、64 個の Player Status Register(PSR)と、4096 個の General Purpose Register (GPR)とからなる。Player Status Register の設定値(PSR)のうち、PSR4~PSR8 は、現在の再生時点を表現するのに用いられる。

PSR4 は、1~100 の値に設定されることで、現在の再生時点が属するタイトルを示し、0 に設定されることで、現在の再生時点がトップメニューであることを示す。

10 PSR5 は、1~999 の値に設定されることで、現在の再生時点が属するチャプター番号を示し、0xFFFF に設定されることで、再生装置においてチャプター番号が無効であることを示す。

PSR6 は、0~999 の値に設定されることで、現在の再生時点が属する PL(カレント PL)の番号を示す。

15 PSR7 は、0~255 の値に設定されることで、現在の再生時点が属する Play Item(カレント Play Item)の番号を示す。

PSR8 は、0~0xFFFFFFFF の値に設定されることで、45KHz の時間精度を用いて現在の再生時点(カレント PTM(Presentation TiMe))を示す。以上の PSR4~PSR8 により、現在の再生時点が特定されることになる。

20 ローカルメモリ 2 9 は、BD-ROM からの読み出しは低速である故、BD-ROM の記録内容を一時的に格納しておくためのキャッシュメモリである。かかるローカルメモリ 2 9 が存在することにより、BD-J モードにおけるアプリケーション実行は、効率化されることになる。図 1 7 (a) は、BD-ROM に存在している Java アーカイブファイルを、ローカルメモリ 2 9 上でどのように識別
25 するかを示す図である。図 1 7 (a) の表は、左欄に BD-ROM 上のファイル名を、右欄にローカルメモリ 2 9 上のファイル名をそれぞれ示している。これら右欄、左欄を比較すれば、ローカルメモリ 2 9 におけるファイルは、ディレクトリ指定"BDJA"を省いたファイルパスで指定されていることがわかる。

図 1 7 (b) は、図 1 7 (a) の応用を示す図である。本応用例は、ヘッダ

+データという形式で、ファイルに格納されているデータを格納するというものである。何をヘッダに用いるかということ、ローカルメモリ29におけるファイルパスを用いる。図17(b)に示したように、ローカルメモリ29ではBD-ROMにおけるファイルパスの一部を省略したものをファイルパスに用いるから、当該ファイルパスをヘッダに格納することで、各データにおけるBD-ROM上の所在を明らかにすることができる。

以上が、本実施形態に係る再生装置のハードウェア構成である。続いて本実施形態に係る再生装置のソフトウェア構成について説明する。

図18は、ROM24に格納されたソフトウェアと、ハードウェアとからなる部分を、レイア構成に置き換えて描いた図である。本図に示すように、再生装置のレイア構成は、以下のa),b),c),d-1),d-2),e),f)からなる。つまり、

- a)物理的なハードウェア階層の上に、
- b)AVClipによる再生を制御するPresentation Engine 31、
- c)プレイリスト情報及びClip情報に基づく再生制御を行うPlayback Control Engine 32、
- という2つの階層があり、
最上位の階層に
- e)タイトル間の分岐を実行するモジュールマネージャ34がある。
- これらHDMVモジュール33、モジュールマネージャ34の間に、
- d-1)Movieオブジェクトの解読・実行主体であるHDMVモジュール33と、
- d-2)BD-Jオブジェクトの解読・実行を行うBD-Jモジュール35とが同じ階層に置かれている。

BD-Jモジュール35は、いわゆるJavaプラットフォームであり、ワークメモリ37を含むJava仮想マシン38を中核にした構成になっていて、アプリケーションマネージャ36、Event Listener Manager 39、Default Operation Manager 40から構成される。先ず初めに、Presentation Engine 31～モジュールマネージャ34について説明する。図19は、Presentation Engine 31～モジュールマネージャ34による処理を模式化した図である。

Presentation Engine 31は、AV再生機能を実行する。再生装置のAV再生

機能とは、DVD プレーヤ、CD プレーヤから踏襲した伝統的な機能群であり、再生開始(Play)、再生停止(Stop)、一時停止(Pause On)、一時停止の解除(Pause Off)、Still 機能の解除(still off)、速度指定付きの早送り(Forward Play(speed))、速度指定付きの巻戻し(Backward Play(speed))、音声切り換え(Audio Change)、
5 副映像切り換え(Subtitle Change)、アングル切り換え(Angle Change)といった機能である。AV 再生機能を実現するべく、Presentation Engine 3 1 は、リードバッファ 2 上に読み出された AVClip のうち、所望に時刻にあたる部分のデコードを行うよう、ビデオデコーダ 4、P-Graphics デコーダ 9、I-Graphics デコーダ 1 3、オーディオデコーダ 2 0 を制御する。所望の時刻として
10 PSR8(カレント PTM)に示される箇所のデコードを行わせることにより、AVClip において、任意の時点を再生を可能することができる。図中の◎1 は、Presentation Engine 3 1 によるデコード開始を模式化して示す。

再生制御エンジン(Playback Control Engine(PCE)) 3 2 は、プレイリストの再生機能(i)、再生装置における状態取得／設定機能(ii)といった諸機能を実行する。PL の再生機能とは、Presentation Engine 3 1 が行う AV 再生機能のうち、再生開始や再生停止を、カレント PL 情報及び Clip 情報に従って行わせることをいう。これら機能(i)～(ii)は、HDMV モジュール 3 3～BD-J モジュール 3 5 からのファンクションコールに応じて実行する。つまり再生制御エンジン 3 2 は、ユーザ操作による指示、レイヤモデルにおける上位層からの指示に応じて、
15 自身の機能を実行する。図 1 9 において、◎2、◎3 が付された矢印は、Clip 情報及びプレイリスト情報に対する Playback Control Engine 3 2 の参照を模式的に示す。

HDMV モジュール 3 3 は、MOVIE モードの実行主体であり、モジュールマネージャ 3 4 から分岐先を構成する Movie オブジェクトが通知されれば、分岐
25 先タイトルを構成する Movie オブジェクトをローカルメモリ 2 9 に読み出して、この Movie オブジェクトに記述されたナビゲーションコマンドを解読し、解読結果に基づき Playback Control Engine 3 2 に対するファンクションコールを実行する。図 1 9 において▽2、▽3、▽4 が付された矢印は、モジュールマネージャ 3 4 からの分岐先 Movie オブジェクトの通知(2)、Movie オブジェクト

に記述されたナビゲーションコマンドの解釈(3)、Playback Control Engine 3 2 に対するファンクションコール(4)を、模式的に示している。

5 モジュールマネージャ 3 4 は、BD-ROM から読み出された Index Table を保持して、分岐制御を行う。この分岐制御は、JumpTitle コマンドを HDMV モジュール 3 3 が実行した場合、又は、タイトルジャンプ API が BD-J モジュール 3 5 からコールされた場合、そのジャンプ先となるタイトル番号を受け取って、そのタイトルを構成する Movie オブジェクト又は BD-J オブジェクトを HDMV モジュール 3 3 又は BD-J モジュール 3 5 に通知するというものである。図中の▽0,▽1,▽2 が付された矢印は、JumpTitle コマンドの実行(0)、モ
10 ジュールマネージャ 3 4 による IndexTable 参照(1)、分岐先 Movie オブジェクト(2)の通知を模式的に示している。

以上が Presentation Engine 3 1 ~モジュールマネージャ 3 4 についての説明である。続いてアプリケーションマネージャ 3 6 について、図 2 0 を参照しながら説明する。図 2 0 は、アプリケーションマネージャ 3 6 を示す図である。

15 アプリケーションマネージャ 3 6 は、アプリケーション管理テーブルを参照したアプリケーションの起動制御、タイトルの正常終了時における制御を実行する。

起動制御とは、モジュールマネージャ 3 4 から分岐先となる BD-J オブジェクトが通知される度に、その BD-J オブジェクトを読み出し、その BD-J オブ
20 ジェクト内のアプリケーション管理テーブルを参照してローカルメモリ 2 9 アクセスを行う。そして現在の再生時点を生存区間とするアプリケーションを構成する xlet プログラムを、ワークメモリに読み出すという制御である。図 2 0 における☆1,☆2,☆3 は、起動制御における分岐先 BD-J オブジェクトの通知(1)、アプリケーション管理テーブル参照(2)、Java 仮想マシン 3 8 に対する起動指
25 示を模式化して示す。この起動指示により Java 仮想マシン 3 8 は、ローカルメモリ 2 9 からワークメモリ 3 7 に xlet プログラムを読み出す(☆5)。

タイトルの終了制御には、正常終了時の制御と、異常終了時の制御とがある。正常終了時の制御には、タイトルを構成するアプリケーションによりジャンプタイトル API がコールされて、分岐先タイトルへの切り換えを分岐制御の主体

(モジュールマネージャ 34)に要求するという制御がある。この終了制御における、モジュールマネージャ 34 通知を模式化して示したのが矢印☆6 である。ここでタイトルを正常終了するにあたって、タイトルを構成するアプリケーションは、起動されたままであってもよい。何故なら、アプリケーションを終了するか否かは、分岐先タイトルにおいて判断するからである。本実施形態では深く触れないが、アプリケーションマネージャ 36 は、BD-ROM からローカルメモリ 29 に Java アーカイブファイルを読み出す(8)との処理を行う。このローカルメモリ 29 への読み出しを模式化したのが☆8 である。

5 以上がアプリケーションマネージャ 36 についての説明である。続いてワークメモリ 37 ~Default Operation Manager 40 について、図 21 を参照しながら説明する。

ワークメモリ 37 は、アプリケーションを構成する xlet プログラムが配置されるヒープ領域である。ワークメモリ 37 は、本来 Java 仮想マシン 38 内に存在するが、図 21 では、作図の便宜上、ワークメモリ 37 を Java 仮想マシン 38 上位層に記述している。ワークメモリ 37 上の xlet プログラムには、EventListner や、JMF プレーヤインスタンスが含まれる。

Java 仮想マシン 38 は、アプリケーションを構成する xlet プログラムをワークメモリ 37 にロードして、xlet プログラムを解読し、解読結果に従った処理を実行する。上述したように xlet プログラムは、JMF プレーヤインスタンス生成を命じるメソッド、この JMF プレーヤインスタンスの実行を命じるメソッドを含むので、これらのメソッドで命じられた処理内容を実現するよう、下位層に対する制御を行う。JMF プレーヤインスタンス生成が命じられれば、Java 仮想マシン 38 は、BD-ROM 上の YYYY.MPLS ファイルに関連付けられた JMF プレーヤインスタンスを得る。また、JMF プレーヤインスタンスにおける JMF メソッドの実行が命じられれば、この JMF メソッドを BD ミドルウェアに発行して、BD 再生装置が対応しているファンクションコールに置き換させる。そして置換後のファンクションコールを Playback Control Engine 32 に発行する。

Event Listner Manager 39 は、ユーザ操作により生じたイベント(キーイベ

ント)を解析し、イベントの振り分けを行う。図中の実線矢印◇1、◇2 は、この Event Listner Manager 3 9 による振り分けを模式的に示す。START、STOP、SPEED 等、xlet プログラム内の Event Listner に登録されたキーイベントなら、BD-J オブジェクトにより間接参照されている xlet プログラムにかかるイベントを振り分ける。START、STOP、SPEED は、JMF に対応したイベントであり、xlet プログラムの Event Listner には、これらのキーイベントが登録されているので、本キーイベントにより xlet プログラムの起動が可能になる。キーイベントが Event Listner 非登録のキーイベントである場合、本キーイベントを Default Operation Manager 4 0 に振り分ける。音声切り換え、アングル切り換え等、BD-ROM 再生装置において生じるキーイベントには、Event Listner に登録されていない多様なものがあり、これらのキーイベントが生じたとしても、漏れの無い処理を実行するためである。

Default Operation Manager 4 0 は、xlet プログラム内の Event Listner に登録されてないキーイベントが Event Listner Manager 3 9 から振り分けられると、その Event Listner 非登録イベントに対応するファンクションコールを Playback Control Engine 3 2 に対して実行する。この Default Operation Manager 4 0 によるファンクションコールを模式的に示したのが、図中の矢印◇3 である。尚、図 2 1 において Event Listner 非登録イベントは Event Listner Manager 3 9、Default Operation Manager 4 0 により振り分けられたが、Playback Control Engine 3 2 がダイレクトに Event Listner 非登録イベントを受け取り、再生制御を行ってもよい(図中の◇4)。

(フローチャートの説明)

以上のアプリケーションマネージャ 3 6 についての説明は、その概要に触れたに過ぎない。アプリケーションマネージャ 3 6 の処理を更に詳しく示したのが図 2 2、図 2 3 のフローチャートである。以降、これらのフローチャートを参照してアプリケーションマネージャ 3 6 の処理手順についてより詳しく説明する。

図 2 2 は、アプリケーションマネージャ 3 6 による分岐時の制御手順を示す

図である。本フローチャートは、ステップS 2～ステップS 5がなす条件を満たすアプリケーション(アプリケーション x という)を、起動又は終了させるという処理である。

5 ステップS 2は、分岐元タイトルで非起動だが、分岐先タイトルで生存して
いて、分岐先タイトルにおける起動属性が AutoRun 属性のアプリケーション x
が存在するか否かの判定であり、もしあれば、ローカルメモリ 29 に対するキ
ャッシュセンスを行う。キャッシュセンスの結果、アプリケーション x がロー
カルメモリ 29 上に有れば(ステップS 7で Yes)、ローカルメモリ 29 からワー
クメモリ 37 にアプリケーション x を読み込む(ステップS 8)。ローカルメモ
10 リ 29 に無ければ、BD-ROM からローカルメモリ 29 にアプリケーション x
を読み込んだ上で、ローカルメモリ 29 からワークメモリ 37 にアプリケー
ション x を読み込む(ステップS 9)。

 ステップS 3は、分岐元タイトルで起動中で、分岐先タイトルで非生存のア
プリケーション x が存在するかどうかの判定である。もし存在するのなら、ア
15 プリケーション x をワークメモリ 37 から削除して終了させる(ステップS 1
0)。

 ステップS 4は、分岐元 Suspend、分岐先 AutoRun 又は Persistent のア
プリケーションが存在するか否かの判定である。もし存在するなら、アプリケー
ション x を Resume する(ステップS 11)。

20 ステップS 5は、分岐元タイトルで起動中で、分岐先 Suspend のアプリケー
ションが存在するか否かの判定である。もし存在すれば、アプリケーション x
を Suspend する(ステップS 12)。

 個々のアプリケーションを終了させるにあたってのアプリケーションマネー
ジャ 36 の処理は、図 23 に示すものとなる。図 23 は、アプリケーション終
25 了処理の処理手順を示すフローチャートである。本図は、終了すべき複数ア
プリケーションのそれぞれについて、ステップS 16～ステップS 20の処理を
繰り返すループ処理になっている(ステップS 15)。本ループ処理においてア
プリケーションマネージャ 36 は起動中アプリケーションを終了するような
terminate イベントを発行し(ステップS 16)、タイマセットして(ステップS

17)、ステップS18～ステップS20からなるループ処理に移行する。この terminate イベントを Event Listener が受信すれば、対応する xlet プログラムは、終了プロセスを起動する。終了プロセスが終了すれば、その xlet プログラムはワークメモリ37から解放され、終了することになる。

- 5 ステップS18～ステップS20のループ処理の継続中、タイマはカウントダウンを継続する。本ループ処理においてステップS18は、発行先アプリケーションが終了したか否かの判定であり、もし終了していればこのアプリケーションに対する処理を終える。ステップS19は、タイマがタイムアウトしたか否かの判定であり、タイムアウトすれば、ステップS20において発行先アプリケーションをワークメモリ37から削除してアプリケーションを強制終了する。

以上のモジュールマネージャ34の処理を、図24を参照しながら説明する。

- 図24は、アプリケーション終了の過程を模式的に示した図である。本図における第1段目は、アプリケーションマネージャ36を、第2段目は、3つの
15 アプリケーションを示す。図24の第2段目、左側のアプリケーションは、 terminate イベントを受信して終了プロセスに成功したアプリケーションを示す。図24の第2段目、中列のアプリケーションは、 terminate イベントを受信したが終了プロセスに失敗したアプリケーションを示す。第2段目、右側のアプリケーションは、 EventListener が実装されていないので、 terminate イベントを受信することができなかったアプリケーションを示す。

第1段目～第2段目間の矢印 ep1,ep2 は、アプリケーションマネージャによる terminate イベント発行を模式的に示し、矢印 ep3 は、終了プロセス起動を模式的に示している。

- 第3段目は、終了プロセス成功時における状態遷移後の状態であり、このアプリケーションは、自身の終了プロセスにより終了することになる。これら xlet
25 プログラムのように、所定の期間内に終了しないアプリケーションがあれば、アプリケーションマネージャ36は、それらを強制的にワークメモリ37から取り除く。第4段目は、アプリケーションマネージャ36による強制終了を示す。かかる第4段目の強制終了を規定するのも、アプリケーションマネージャ

36の1つの使命といえる。

- 5 以上のように本実施形態によれば、分岐元タイトルで起動しており、分岐先タイトルで生存していないアプリケーションは、自動的に終了させられるので、条件付き分岐により再生が複雑に進行する場合でも、再生装置におけるリソースの限界を越える数のアプリケーション立ち上げはなされ無い。分岐前後におけるアプリケーション動作を保証することができるので、デジタルストリームを再生させながら、アプリケーションを実行させるようなディスクコンテンツを多く頒布することができる。

(第2実施形態)

- 10 第1実施形態においてアプリケーションの生存区間は、タイトル時間軸と一致していたが、第2実施形態は、PL 時間軸の一部をアプリケーションの生存区間とすることを提案する。PL 時間軸の一部は、チャプターにより表現されるので、チャプターにて開始点、終了点を記述することにより、アプリケーションの生存区間を規定することができる。図25(a)は、PL 時間軸上に生存区間を定めたアプリケーション管理テーブルを示す図である。図25(a)においてアプリケーション管理テーブルには、3つのアプリケーションが記述
- 15 されており、このうち application#2 は、title#1 の Chapter#2 から Chapter#3 までが生存区間に指定され、起動属性に AutoRun が規定されている。このため application#2 は、図25(b)に示すように、Chapter#2 の始点で起動され、Chapter#3 の終点で終了することになる。
- 20

一方 application#3 は、title#1 の Chapter#4 から Chapter#6 までが生存区間に指定されている。このため application#3 は、図25(b)に示すように、図25(b)に示すように、Chapter#4 の始点で起動され、Chapter#6 の終点で終了することになる。

- 25 こうして記述されたアプリケーション管理テーブルに基づき、処理を行うため本実施形態に係るアプリケーションマネージャ36は、PLmarkにより指示されるチャプター開始点に到達する度に、そのチャプター開始点から生存区間が始まるアプリケーションが存在するか否かを判定し、もし存在すればそのアプリケーションをワークメモリ37にロードする。

同様に、チャプター開始点に到達する度に、そのチャプターの直前のチャプターで生存区間が終わるアプリケーションが存在するか否かを判定し、もし存在すればそのアプリケーションをワークメモリ 37 から解放する。

- 5 チャプターという単位でアプリケーションの生存を管理すれば、アプリケーションの生存区間をより細かい精度で指定することができる。しかしディスクコンテンツには、時間軸の逆向がありうることに留意せねばならない。逆行とは、巻戻しにより時間軸を逆向きに進行することである。チャプターの境界でこの逆行と、進行とが繰り返されれば、ワークメモリへのロード、廃棄が何度もなされ、余分な読出負荷が生じる。そこで本実施形態では、アプリケーションの起動時期を、タイトルに入って Playback Control Engine 32 による通常再生が開始された瞬間にしている。ここで PL の再生には、通常再生、トリック再生がある。トリック再生とは、早送り、巻戻し、SkipNext, SkipBack がある。かかる、早送り、巻戻し、SkipNext, SkipBack がなされている間、アプリケーション起動を開始せず、通常再生が開始されて初めて、アプリケーションを起動するのである。通常再生開始の瞬間を基準にすることにより、上述したような生存区間前後の行き来があった場合でも、アプリケーションの起動が必要以上に繰り返されることはない。尚、通常再生開始の瞬間を、アプリケーションの起動基準にするとの処理は、生存区間が title である場合にも、実行してよい。
- 10 以上のように本実施形態によれば、PL より小さい、チャプターの単位でアプリケーションの生存区間を規定することができるので、緻密なアプリケーション制御を実現することができる。

(第 2 実施形態の変更例)

- 25 図 25 では、各アプリケーションに優先度が付与されている。この優先度は、0~255 の値をとり、アプリケーション間でリソースの使用が競合等が競合した場合、どちらのアプリケーションを強制的に終了させるか、また、どちらのアプリケーションからリソースを奪うかという処理をアプリケーションマネージャ 36 が行うにあたって、判断材料になる。図 25 の一例では、application#1 の優先度は 255, application#2、application#3 の優先度は 128 なので、

application#1-application#2 の競合時において、アプリケーションマネージャ 36 は優先度が低い application#2 を強制終了するとの処理を行う。

(第 3 実施形態)

BD-ROM により供されるディスクコンテンツは、互いに分岐可能な複数タイトルから構成される。各タイトルは、1 つ以上の PL と、この PL を用いた制御手順とからなるもの以外に、再生装置に対する制御手順のみからなる非 AV 系タイトルがある。本実施形態は、この非 AV 系タイトルについて説明する。

こうした非 AV 系タイトルのタイトルでは、どのようにタイトル時間軸を定めるのかが問題になる。図 26 (a) は、PL 時間軸から定まるタイトル時間軸を示す。この場合 PL 時間軸がタイトル時間軸になり、このタイトル時間軸上にアプリケーションの生存区間が定まる。この基準となる PL 時間軸がない場合、タイトル時間軸は図 26 (b) (c) のように定めるべきである。

図 26 (b) は、メインとなるアプリケーションの生存区間から定まるタイトル時間軸を示す。メインアプリとは、タイトルにおいて起動属性が AutoRun に設定され、タイトル開始時に自動起動される唯一のアプリケーションであり、例えばランチャーアプリと呼ばれるものがこれにあたる。ランチャーアプリとは、他のアプリケーションを起動するアプリケーションプログラムである。

この図 26 (b) の考え方は、メインアプリが起動している限り、タイトル時間軸は継続していると考え、メインアプリが終了すれば、時間軸を終結させるというものである。図 26 (c) は、複数アプリケーションの生存区間から定まるタイトル時間軸を示す図である。タイトルの開始点で起動されるのは 1 つのアプリケーションであるが、このアプリケーションが他のアプリケーションを呼び出し、更にこのアプリケーションが別のアプリケーションを呼び出すとの処理が繰り返されるというケースがある。この場合、どれかのアプリケーションが起動している限り、タイトル時間軸は継続していると考え、どのアプリケーションも起動していない状態が到来すれば、そこでタイトル時間軸は終結するという考え方である。このように非 AV 系タイトルのタイトル時間軸を定めれば、AV タイトルであっても、非 AV 系タイトルであっても、タイトル時間軸の終結と同時に、所定のタイトルに分岐するという処理を画一的に行うこ

とができる。尚非 AV タイトルにおけるタイトル時間軸は、AV タイトルと対比する上で、想定した架空の時間軸に過ぎない。故に再生装置は、非 AV タイトルにおけるタイトル時間軸上を逆行したり、任意の位置に頭出しすることができない。

- 5 以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。

上述したような手順でタイトル終了を行うため第3実施形態に係るアプリケーションマネージャ36は、図27に示すような処理で処理を行う。図27は、タイトル再生時におけるアプリケーションマネージャ36の処理手順を示すフローチャートである。本フローチャートは、タイトル再生中、ステップS21
10 〜ステップS23を繰り返すというループ構造になっている。

ステップS21は、タイトルジャンプAPIが呼び出されたか否かの判定であり、呼び出されれば、ジャンプ先タイトルへの分岐をモジュールマネージャ34に要求する(ステップS27)。

- 15 ステップS22は、タイトル内のアプリケーション呼出を担っているようなメインアプリが存在するか否かの判定であり、もし存在するなら、その起動の有無を確認する(ステップS25)。起動してなければ、“タイトルの終わり”であると解釈し、モジュールマネージャ34に終結を通知する(ステップS26)。

- 20 ステップS23は、メインアプリがない場合に実行されるステップであり(ステップS22でNo)、どのアプリケーションも起動していない状態かどうかを判定する。もしそうなら、同じく“タイトルの終わり”であると解釈し、モジュールマネージャ34に終結を通知する(ステップS26)。

- 25 以上のように本実施形態によれば、PL再生を伴わないタイトルであとしても、アプリケーション実行中は分岐せず、アプリケーション実行が終了して初めて分岐するという処理が可能になる。

(第4実施形態)

本実施形態は、DVDと同様のメニュー制御をBD-ROM上で実現する場合の改良に関する。図28(a)は、BD-ROMにより実現されるメニュー階層を

示す図である。本図におけるメニュー階層は、TopMenu を最上位に配し、この TopMenu から下位の TitleMenu、SubTitleMenu、AudioMenu を選択できる構造になっている。図中の矢印 sw1,2,3 は、ボタン選択によるメニュー切り換えを模式的に示す。TopMenu とは、音声選択、字幕選択、タイトル選択の
5 何れを行うかを受け付けるボタン(図中のボタン sn1,sn2,sn3)を配置したメニューである。

TitleMenu とは、映画作品(title)の劇場版を選択するか、ディレクターズカット版を選択するか、ゲーム版を選択するか等、映画作品の選択を受け付けるボタンを配置したメニューである。AudioMenu とは、音声再生を日本語で行
10 うか、英語で行うかを受け付けるボタンを配置したメニュー、SubTitleMenu とは、字幕表示を日本語で行うか、英語で行うかを受け付けるボタンを配置したメニューである。

こうした階層をもったメニューを動作させるための MOVIE オブジェクトを図 28 (b) に示す。図 28 (b) において MovieObject.bdmv には、FirstPlay
15 OBJ、TopMenu OBJ、AudioMenu OBJ、SubTitleMenu OBJ が格納されている。

FirstPlay オブジェクト(FirstPlay OBJ)は、再生装置への BD-ROM のローディング時に自動的に実行される動的シナリオである。

TopMenu オブジェクト(TopMenu OBJ)は、TopMenu の挙動を制御する動的シナリオである。ユーザがメニューコールを要求した際、呼び出されるのはこの TopMenu オブジェクトである。TopMenu オブジェクトは、ユーザからの操作に応じて TopMenu 中のボタンの状態を変えるものや、ボタンに対する確定
20 操作に応じて分岐を行う分岐コマンドを含む。この分岐コマンドは、TopMenu から TitleMenu、TopMenu から SubTitleMenu、TopMenu から AudioMenu
25 というメニュー切り換えを実現するものである。

AudioMenu オブジェクト(AudioMenu OBJ)は、AudioMenu の挙動を制御する動的シナリオであり、ユーザからの操作に応じて AudioMenu 中のボタンの状態を変えるコマンドや、ボタンに対する確定操作に応じて音声設定を更新するコマンドを含む。

SubTitleMenu オブジェクト(SubTitleMenu OBJ)は、SubTitleMenu の挙動を制御する動的なシナリオであり、ユーザからの操作に応じて SubTitleMenu 中のボタンの状態を変えるコマンドや、ボタンに対する確定操作に応じて字幕設定用の PSR を更新するコマンドを含む。

- 5 TitleMenu オブジェクト(TitleMenu OBJ)は、TitleMenu の挙動を制御する動的シナリオであり、TitleMenu 中のボタンの状態を変えるものや、ボタンに対する確定操作に応じて分岐を行う分岐コマンドをふくむ。

- これらのメニュー用 MOVIE オブジェクトにより、DVD で実現されているようなメニューの挙動を実現することができる。以上がメニュー制御に関連する MOVIE オブジェクトである。

- 図29は、Index Table と、Index Table から各 Movie オブジェクトへの分岐とを模式化した図である。本図では左側に Index Table の内部構成を示している。本実施形態における Index Table には、FirstPlayINDEX、TopMenuINDEX、Audio MenuINDEX、Subtitle MenuINDEX、title MenuINDEX、title#1～#mINDEX、title#m+1～#nINDEX、title#0INDEX を含む。図中の矢印 bc1,2 は、Index Table から FirstPlayOBJ への分岐と、FirstPlayOBJ から TopMenu への分岐とを模式的に示し、矢印 bc3,4,5 は、TopMenu から TitleMenu、SubTitleMenu、AudioMenu への分岐を模式的に示している。矢印 bc6,7,8 は、TitleMenu から各 Movie オブジェクトへの分岐を模式的に示している。

FirstPlayINDEX、TopMenuINDEX、Audio MenuINDEX、Subtitle MenuINDEX、title MenuINDEX は、それぞれ、FirstPlayOBJ、TopMenuOBJ、Audio MenuOBJ、Subtitle MenuOBJ、title MenuOBJ についての Index であり、これらの識別子が記述される。

- 25 title#1～#mINDEX は、BD-ROM において 1 から m 番目にエンタリーされている title についての Index であり、これら 1 から m までの title 番号の選択時において分岐先となる MOVIE オブジェクトの識別子(ID)が記述される。

title#m+1～#nINDEX は、BD-ROM において m+1 から n 番目にエンタリーされている title についての Index であり、これら m+1 から n までの title

番号の選択時において分岐先となる BD-J オブジェクトの識別子(ID)が記述される。

5 title#0INDEX は、BD-J オブジェクトの強制終了時において分岐先になるべき Movie オブジェクト又は BD-J オブジェクトを規定する INDEX である。本実施形態では、TopMenuOBJ についての識別子が、この title#0INDEX に格納されている。

10 図30(a)は、図29のように Index Table が記述された場合における分岐を示す。Index Table がこのように記述されているので、ラベル title#1～title#m を分岐先とした分岐コマンドの実行時には、title#1Index～title#mIndex から Movie オブジェクト#1～#m の識別子が取り出される。ラベル title#m+1～title#n を分岐とした分岐コマンドの実行時には、title#m+1Index～title#nIndex から BD-J オブジェクト#m+1～#n の識別子が取り出される。BD-J オブジェクト#m+1～#n の識別子は、ファイル名を表す5桁の数値であるので、『00001.BD-J,00002.BD-J,00003.BD-J・・・』が取り出され、そのファイル名の動的シナリオがメモリに読み出されて、実行されることになる。これが Index Table を用いた分岐処理である。

20 図30(b)は、BD-J オブジェクト実行時の強制終了時における分岐を示す図である。強制終了時における分岐では、title#0Index から識別子が取り出されて、その識別子の動的シナリオが再生装置により実行される。この識別子が、トップメニュータイトルの識別子なら、アプリケーション強制終了時には、自動的にトップメニューOBJ が選択されることになる。

25 以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。上述した記録媒体の改良に対応するため、再生装置内のモジュールマネージャ34は図31に示すような処理手順で処理を行う。図31は、モジュールマネージャ34の処理手順を示すフローチャートである。本フローチャートは、ステップS31、ステップS32からなるループ処理を構成しており、ステップS31又はステップS32のどちらかが Yes になった際、対応する処理を実行するものである。

ステップS 3 1 は、タイトルジャンプ API の呼び出しがあったか否かの判定である。もしタイトルジャンプ API の呼び出しがあれば、分岐先ラベルであるタイトル番号 j を取得し(ステップS 3 3)、Index Table におけるタイトル番号 j の Index から、IDj を取り出して(ステップS 3 4)、IDj の Movie オブジェクト又は BD-J オブジェクトを、HDMV モジュール 3 3 又は BD-J モジュール 3 5 に実行させる(ステップS 3 5)。

ステップS 3 2 は、タイトル終了がアプリケーションマネージャ 3 6 から通知されたか否かの判定であり、もし通知されれば(ステップS 3 2 で Yes)、トップメニュータイトルを構成するトップメニューOBJ を HDMV モジュール 3 3 又はモジュールマネージャ 3 4 に実行させる(ステップS 3 6)。

以上のアプリケーションマネージャ 3 6 によるアプリケーション強制終了の動作例を、図 3 2 を参照しながら説明する。ここで再生すべきタイトルは、落下するタイル片を積み重ねるというゲームアプリを含む非 AV 系タイトルである。図 3 2 の下段は、アプリケーションの生存区間からなるタイトル時間軸を示し、上段は、タイトル時間軸において表示される画像を示す。非 AV 系タイトルがゲームアプリである場合、このゲームアプリの生存区間において、図 3 2 の上段左側のように、ゲームアプリの一画面が表示される。ゲームアプリにバグがあり、異常終了すると、アプリケーションマネージャ 3 6 は図 2 3 のフローチャートに従ってゲームアプリを強制終了させ、タイトルの終了をモジュールマネージャ 3 4 に通知する。タイトル終了が通知されると、モジュールマネージャ 3 4 はトップメニュータイトルに分岐する。そうすると、図 3 2 の上段右側に示すような画像が表示され、ユーザの操作待ちになる。

以上のように本実施形態によれば、プログラムが含むが、デジタルストリームは含まないような非 AV 系タイトルの終了時においても、トップメニュータイトルに分岐するという制御が可能になる。これによりアプリケーションプログラムがエラー終了したとしても、ブラックアウトやハングアップの発生を回避することができる。

(第 5 実施形態)

BD-J モードにおいて、PL 再生との同期をどのように実現するかという改良に関する。図 8 (b) の一例において JMF プレーヤインスタンスの再生を命じる JMF プレーヤインスタンス(A.play;)を Java 仮想マシン 3 8 が解読した場合、Java 仮想マシン 3 8 は PL 再生 API をコールして、コール直後に” サクセス” を示す応答をアプリケーションに返す。

Playback Control Engine 3 2 は、PL 再生 API がコールされれば、PL 情報に基づく処理手順を実行する。PL が 2 時間という再生時間を有するなら、この 2 時間の間、上述した処理は継続することになる。ここで問題になるのは、Java 仮想マシン 3 8 がサクセス応答を返す時間と、Playback Control Engine 3 2 が実際に処理を終える時間とのギャップである。Java 仮想マシン 3 8 は、イベントドリブンの処理主体であるためコール直後に再生成功か、再生失敗かを示す応答を返すが、Playback Control Engine 3 2 による実際の処理終了は 2 時間経過後であるので、サクセス応答をアプリケーションに返す時間を基準にしたのでは、2 時間経過後にあたる処理終結を感知しえない。PL 再生において早送り、巻戻し、Skip が行われると、この 2 時間という再生期間は 2 時間前後に変動することになり、処理終結の感知は更に困難になる。

Playback Control Engine 3 2 は、アプリケーションとスタンドアローンで動作するため、第 3 実施形態のような終了判定では、PL 再生の終了をタイトル終了と解釈することができない。そこで本実施形態では、アプリケーションが終了してようがいまいが、ワークメモリ 3 7 に JMF プレーヤインスタンスがある限り、つまり、Presentation Engine 3 1 の制御権を BD-J モジュール 3 5 が掌握している間、Playback Control Engine 3 2 から再生終結イベントを待つ。そして再生終結イベントがあれば、タイトルが終了したと解釈して、次のタイトルへの分岐を行うようモジュールマネージャ 3 4 に通知する。こうすることにより、Playback Control Engine 3 2 が PL 再生を終結した時点、タイトルの終端とすることができる。

以降図 3 3 ~ 図 3 7 のフローチャートを参照して、Playback Control Engine 3 2 による具体的な制御手順を説明する。

図 3 3 は、Playback Control Engine 3 2 による PL 再生手順を示すフローチ

- ャートである。この再生手順は、Presentation Engine 3 1 に対する制御(ステップ S 4 6)と、BD-ROM ドライブ 1 又は HDD 1 7 に対する制御(ステップ S 4 8)とを主に含む。本フローチャートにおいて処理対象たる PlayItem を PlayItem#x とする。本フローチャートは、カレント PL 情報(.mpls)の読み込みを行い(ステップ S 4 1)、その後、ステップ S 4 2～ステップ S 5 0 の処理を実行するというものである。ここでステップ S 4 2～ステップ S 5 0 は、ステップ S 4 9 が Yes になるまで、カレント PL 情報を構成するそれぞれの PI 情報について、ステップ S 4 3～ステップ S 5 0 の処理を繰り返すというループ処理を構成している。このループ処理において処理対象となる PlayItem を、
- 10 PlayItem#x(PI#x)とよぶ。この PlayItem#x は、カレント PL の先頭の PlayItem に設定されることにより、初期化される(ステップ S 4 2)。上述したループ処理の終了要件は、この PlayItem#x がカレント PL の最後の PlayItem になることであり(ステップ S 4 9)、もし最後の PlayItem でなければ、カレント PL における次の PlayItem が PlayItem#x に設定される(ステップ S 5 0)。
- 15 ループ処理において繰り返し実行されるステップ S 4 3～ステップ S 5 0 は、PlayItem#x の Clip_information_file_name で指定される Clip 情報をシナリオメモリ 2 1 に読み込み(ステップ S 4 3)、PlayItem#x の In_time を、カレント Clip 情報の EPmap を用いて、I ピクチャアドレス u に変換し(ステップ S 4 4)、PlayItem#x の Out_time を、カレント Clip 情報の EP_map を用いて、I ピクチャアドレス v に変換して(ステップ S 4 5)、これらの変換で得られたアドレス v の次の I ピクチャを求めて、そのアドレスの 1 つ手前をアドレス w に設定し(ステップ S 4 7)、そうして算出されたアドレス w を用いて、I ピクチャアドレス u からアドレス w までの TS パケットの読み出しを BD-ROM ドライブ 1 又は HDD 1 7 に命じるというものである(ステップ S 4 8)。
- 20 一方、Presentation Engine 3 1 に対しては、カレント PLMark の mark_time_stamp から PlayItem#x の Out_time までの出力を命じる(ステップ S 4 6)。以上のステップ S 4 5～ステップ S 4 8 により、AVClip において、PlayItem#x により指示されている部分の再生がなされることになる
- その後、PlayItem#x がカレント PL の最後の PI であるかの判定がなされる

(ステップ S 4 9)。

PlayItem#x がカレント PL の最後の PI でなければ、カレント PL における次の PlayItem を、PlayItem#x に設定して(ステップ S 5 0)、ステップ S 4 3 に戻る。以上のステップ S 4 3～ステップ S 5 0を繰り返すことにより、PL
5 を構成する PI は順次再生されることになる。

図 3 4 は、アングル切り換え手順及び SkipBack,SkipNext の手順を示すフローチャートである。本フローチャートは、図 3 3 の処理手順と並行してなされるものであり、ステップ S 5 1～S 5 2からなるループ処理を繰り返すというものである。本ループにおけるステップ S 5 1 は、アングル切り換えを要求
10 する API が、Java 仮想マシン 3 8 からコールされたか否かの判定であり、アングル切り換え API のコールがあれば、カレント Clip 情報を切り換えるという操作を実行する。

図 3 4 のステップ S 5 5 は、判定ステップであり、PlayItem#x の is_multi_angles がオンであるか否かの判定を行う。is_multi_angles とは、
15 PlayItem#x がマルチアングルに対応しているか否かを示すフラグであり、もしステップ S 5 5 が No であるならステップ S 5 3 に移行する。ステップ S 5 5 が Yes であるなら、ステップ S 5 6～ステップ S 5 9を実行する。ステップ S 5 6～ステップ S 5 9は、切り換え後のアングル番号を変数 y に代入して(ステップ S 5 6)、PlayItem#x における y 番目の Clip_information_file_name
20 で指定されている Clip 情報をシナリオメモリ 2 1 に読み出し(ステップ S 5 7)、カレント PTM を、カレント Clip 情報の EP_map を用いて I ピクチャアドレス u に変換し(ステップ S 5 8)、PlayItem#x の Out_time を、カレント Clip 情報の EP_map を用いて I ピクチャアドレス v に変換する(ステップ S 5 9)というものである。こうして I ピクチャアドレス u,v を変化した後、ステップ S
25 4 6に移行する。ステップ S 4 6への移行により、別の AVClip から TS パケットが読み出されるので、映像内容が切り換わることになる。

一方、図 3 4 のループにおけるステップ S 5 2 は、SkipBack/SkipNext を意味する API が Java 仮想マシン 3 8 からコールされたか否かの判定であり、もしコールされれば、図 3 5 のフローチャートの処理手順を実行する。図 3 5 は、

SkipBack, SkipNextAPI がコールされた際の処理手順を示すフローチャートである。SkipBack, SkipNext を実行するにあたっての処理手順は多種多様なものである。ここで説明するのはあくまでも一例に過ぎないことに留意されたい。

5 ステップ S 6 1 は、PSR で示されるカレント PI 番号、及び、カレント PTM を変換することにより、カレント Mark 情報を得る。ステップ S 6 2 は、押下されたのが SkipNext キーであるか、SkipBack キーであるかの判定であり、SkipNext キーであるならステップ S 6 3 において方向フラグを+1 に設定し、SkipBack キーであるならステップ S 6 4 において方向フラグを-1 に設定する。

10 ステップ S 6 5 は、カレント PLMark の番号に方向フラグの値を足した番号を、カレント PLMark の番号として設定する。ここで SkipNext キーであるなら方向フラグは+1 に設定されているのでカレント PLMark はインクリメントされることになる。SkipBack キーであるなら方向フラグは-1 に設定されているので、カレント PLMark はデクリメントされることになる。

15 ステップ S 6 6 では、カレント PLMark の ref_to_PlayItem_Id に記述されている PI を、PlayItem#x に設定し、ステップ S 6 7 では、PlayItem#x の Clip_information_file_name で指定される Clip 情報を読み込む。ステップ S 6 8 では、カレント Clip 情報の EP_map を用いて、カレント PLMark の mark_time_stamp を、I ピクチャアドレス u に変換する。一方ステップ S 6 9 では、PlayItem#x の Out_time を、カレント Clip 情報の EP_map を用いて、I
20 ピクチャアドレス v に変換する。ステップ S 7 0 は、カレント PLMark の mark_time_stamp から PlayItem#x の Out_time までの出力を Presentation Engine 3 1 に命じた上で、図 3 3 のステップ S 4 7 に移行する。こうして I ピクチャアドレス u, v を変化して、別の部分の再生を命じた上でステップ S 4 7 への移行するので、別の AVClip から TS パケットが読み出されることになり、
25 映像内容が切り換えが実現する。

図 3 6 は、Presentation Engine 3 1 による処理手順の詳細を示すフローチャートである。本フローチャートは、I ピクチャの PTS をカレント PTM に設定した後で(ステップ S 7 1)、ステップ S 7 2 ~ ステップ S 7 7 からなるループ処理を実行するものである。

続いてステップ S 7 2 ～ステップ S 7 7 におけるループ処理について説明する。このループ処理は、カレント PTM にあたるピクチャ、オーディオの再生出力と、カレント PTM の更新とを繰り返すものである。本ループ処理におけるステップ S 7 6 は、ループ処理の終了要件を規定している。つまりステップ

5 S 7 6 は、カレント PTM が PI#x の Out_time であることをループ処理の終了要件にしている。

ステップ S 7 3 は、早送り API、又は、早戻し API が Java 仮想マシン 3 8 からコールされたか否かの判定である。もしコールされれば、ステップ S 7 8 において早送りか早戻しかの判定を行い、早送りであるなら、次の I ピクチャ

10 の PTS をカレント PTM に設定する(ステップ S 7 9)。このようにカレント PTM を、次の I ピクチャの PTS に設定することで、1 秒飛びに AVClip を再生してゆくことができる。これにより、2 倍速等で AVClip は順方向に早く再生されることになる。早戻しであるなら、カレント PTM が PlayItem#x の Out_time に到達したかを判定する(ステップ S 8 0)。もし到達してないのなら、

15 1 つ前の I ピクチャの PTS をカレント PTM に設定する(ステップ S 8 1)。このように読出先アドレス A を、1 つ前の I ピクチャに設定することで、AVClip を後方向に 1 秒飛びに再生してゆくことができる。これにより、2 倍速等で AVClip は、逆方向に再生されることになる。尚、早送り、巻戻しを実行するにあたっての処理手順は多種多様なものである。ここで説明するのはあくまで

20 も一例に過ぎないことに留意されたい。

ステップ S 7 4 は、メニューコール API がコールされたか否かの判定であり、もしコールされれば、現在の再生処理をサスペンドして(ステップ S 8 2)、メニュー処理用のメニュープログラムを実行する(ステップ S 8 3)。以上の処理により、メニューメニューコールがなされた場合は、再生処理を中断した上で、

25 メニュー表示のための処理が実行されることになる。

ステップ S 7 5 は、sync_PlayItem_id により、PlayItem#x を指定した SubPlayItem#y が存在するか否かの判定であり、もし存在すれば、図 3 7 のフローチャートに移行する。図 3 7 は、SubPlayItem の再生手順を示すフローチャートである。本フローチャートでは、先ずステップ S 8 6 において、カレン

ト PTM は SubPlayItem#y の sync_start_PTS_of_playItem であるか否かを判定する。もしそうであれば、ステップ S 9 3 において SubPlayItem#y に基づく再生処理を行うよう Playback Control Engine 3 2 に通知する。

図 3 7 のステップ S 8 7 ~ ステップ S 9 2 は、SubPlayItem#y に基づく再生
5 処理を示すフローチャートである。

ステップ S 8 7 では、SubPlayItem#y の Clip_information_file_name で指定される Clip 情報を読み込む。ステップ S 8 8 では、カレント Clip 情報の EP_map を用いて、SubPlayItem#y の In_time を、アドレス α に変換する。一方ステップ S 8 9 では、SubPlayItem#y の Out_time を、カレント Clip 情報の EP_map を用いて、アドレス β に変換する。ステップ S 9 0 は、SubPlayItem#y の In_time から SubPlayItem#y の Out_time までの出力をデコードに命じる。これらの変換で得られたアドレス β の次の I ピクチャを求めて、そのアドレスの 1 つ手前をアドレス γ に設定し(ステップ S 9 1)、そうして算出されたアドレス γ を用いて、SubClip#z におけるアドレス α からアドレス γ までの TS パケットの読み出しを BD-ROM ドライブ 1 又は HDD 1 7 に命
10 じるというものである(ステップ S 9 2)。

また図 3 3 に戻って Playback Control Engine 3 2 の処理の説明の続きを行う。ステップ S 5 3 は Presentation Engine 3 1 による再生制御が完了したかの判定であり、最後の PlayItem#x に対して、図 3 6 のフローチャートの処理が行われている限り、ステップ S 5 3 が No になる。図 3 6 のフローチャートの処理が終了して初めて、ステップ S 5 3 は Yes になりステップ S 5 4 に移行する。ステップ S 5 4 は、Java 仮想マシン 3 8 への再生終結イベントの出力であり、この出力により、2 時間という再生時間の経過を Java 仮想マシン 3 8 は知ることができる。

25 以上が本実施形態における Playback Control Engine 3 2、Presentation Engine 3 1 の処理である。続いて本実施形態におけるアプリケーションマネージャ 3 6 処理手順について説明する。図 3 8 は、第 5 実施形態に係るアプリケーションマネージャ 3 6 の処理手順を示すフローチャートである。

図 3 8 のフローチャートは、図 2 7 のフローチャートを改良したものである。

その改良点は、ステップ S 2 1 からステップ S 2 2 間にステップ S 2 4 が追加され、このステップ S 2 4 が Yes になった際、実行されるステップ S 1 0 1 が存在する点である。

5 ステップ S 2 4 は、JMF プレーヤインスタンスがワークメモリ 3 7 に存在するか否かの判定であり、もし存在しなければステップ S 2 2 に移行する。存在すれば、ステップ S 1 0 1 に移行する。ステップ S 1 0 1 は、Playback Control Engine 3 2 から再生終結イベントが出力されたか否かの判定であり、もし出力されれば、ワークメモリ中の Java プレーヤインスタンスを消滅させた上で(ステップ S 1 0 2)、タイトル終了をモジュールマネージャ 3 4 に通知する(ステップ S 2 6)。通知されねば、ステップ S 2 1 からステップ S 2 4 からのループ処理を繰り返す。

15 以上のフローチャートにおいて、ワークメモリ 3 7 に JMF プレーヤインスタンスが存在する限り(ステップ S 2 4 で Yes)、ステップ S 2 2、ステップ S 2 3 はスキップされる。そのため、たとえ全てのアプリケーションが終了したとしてもタイトルは継続中と解釈される。

 以上のように本実施形態によれば、2 時間という再生時間の経過時点を実アプリケーションマネージャ 3 6 は把握することができるので、PL 再生の終了条件にメニューを表示して、このメニューに対する操作に応じて他のタイトルに分岐するという制御を実現することができる。

20 (第 6 実施形態)

 第 6 実施形態は、BD-J オブジェクトにデータ管理テーブルを設ける改良に関する。

25 データ管理テーブル(DMT)は、そのタイトル時間軸においてローカルメモリ 2 9 上にロードすべき Java アーカイブファイルを、読込属性と、読込優先度とに対応づけて示すテーブルである。”ローカルメモリ 2 9 における生存”とは、そのアプリケーションを構成する Java アーカイブファイルがローカルメモリ 2 9 から読み出され、Java 仮想マシン 3 8 内のワークメモリ 3 7 への転送が可能になっている状態をいう。図 3 9 は、データ管理テーブルの一例を示す図である。本図に示すようにデータ管理テーブルは、アプリケーションの『生存区

間』と、その生存区間をもったアプリケーションを識別する『applicationID』と、そのアプリケーションの『読込属性』と、『読込優先度』とを示す。

上述したようにアプリケーション管理テーブルには、生存区間という概念があり、データ管理テーブルにも同じ生存区間という概念がある。アプリケーション管理テーブルと同じ概念を、データ管理テーブルに設けておくというのは一見無駄のように思えるがこれには意図がある。

図40は、BD-J オブジェクトが想定している実行モデルを示す図である。本図における実行モデルは、BD-ROM、ローカルメモリ29、Java 仮想マシン38からなり、BD-ROM、ローカルメモリ29、ワークメモリ37という三者の関係を示す。矢印 my1 は、BD-ROM→ローカルメモリ29間の読み込みを示し、矢印 my2 は、ローカルメモリ29→ワークメモリ37間の読み込みを示す。矢印上の注釈は、これらの読み込みが、どのようなタイミングでなされるかを示す。注釈によると、BD-ROM→ローカルメモリ29間の読み込みは、いわゆる”先読み”であり、アプリケーションが必要となる以前の時点に行われねばならない。

また注釈によると、ローカルメモリ29→ワークメモリ37間の読み込みは、アプリケーションが必要になった際になされることがわかる。”必要になった際”とは、アプリケーションの生存区間が到来した時点(1)、アプリケーションの呼出が他のアプリケーション又はアプリケーションマネージャ36から指示された時点(2)を意味する。

矢印 my3 は、ワークメモリ37におけるアプリケーションの占有領域の解放を示し、矢印 my4 は、ローカルメモリ29におけるアプリケーションの占有領域の解放を示す。矢印上の注釈は、これらの読み込みが、どのようなタイミングでなされるかを示す。注釈によると、ワークメモリ37上の解放は、アプリケーション終了と同時になされることがわかる。一方ローカルメモリ29上の解放は、Java 仮想マシン38にとって必要でなくなった時点でなされる。この必要でなくなった時点とは、”終了時点”ではない。”終了した上、再起動の可能性もない時点”であること、つまり該当する title が終了した時点の意味する。上述した読込・解放のうち、ワークメモリ37における解放時点は、アプリケー

ション管理テーブルにおける生存区間から判明する。しかし”アプリケーションが必要となる以前の時点”、“終了した上、再起動の可能性もない時点”については、規定し得ない。そこで、オーサリング段階において、かかる時点ディスクコンテンツ全体の時間軸上で規定しておくため、本実施形態では各アプリケーションが生存している区間を、アプリケーション管理テーブルとは別に、データ管理テーブルに記述するようにしている。つまり”アプリケーションが必要となる以前の時点”をデータ管理テーブルにおける生存区間の始点と定義し、“終了した上、再起動の可能性もない時点”をデータ管理テーブルの終点と定義することにより、上述したローカルメモリ29上の格納内容の遷移をオーサリング時に規定しておくことができる。これがデータ管理テーブルの記述意義である。

データ管理テーブルによるローカルメモリ29生存区間の記述について説明する。ここで制作しようとするディスクコンテンツは3つのタイトル(title#1、title#2、title#3)からなり、これらタイトルの時間軸において、図41(b)に示すようなタイミングで、ローカルメモリ29を使用したいと考える。この場合、title#1時間軸の開始点において application#1、application#2 を構成する Java アーカイブファイルをローカルメモリ29に読み込み、title#1時間軸の継続中、application#1、application#2 をローカルメモリ29に常駐させておく。そして title#2時間軸の始点で、application#1 を構成する Java アーカイブファイルをローカルメモリ29から解放して、代わりに application#3 を構成する Java アーカイブファイルをローカルメモリ29に読み込んで、常駐させるというものである(以降、アプリケーションを構成する Java アーカイブファイルは、アプリケーションと同義に扱う。)。この場合のデータ管理テーブルの記述は、図41(a)の通りであり、アプリケーションの applicationID を、その生存区間に対応づけて記述することで、ローカルメモリ29に常駐すべきアプリケーションを表現する。図41(a)では、application#1の applicationID が title#1 と対応づけられて記述されており、application#2の applicationID は title#1、title#2 と対応づけられ、application#3の applicationID は title#3 と対応づけられて記述されていることがわかる。こうすることで、ローカルメ

メモリ 29 占有の時間的遷移がオーサリング担当者により規定されることになる。

データ管理テーブル、アプリケーション管理テーブルの組合せとしては、アプリケーション管理テーブルに規定する生存区間は、細かい再生単位にし、データ管理テーブルに規定する生存区間は、大まかな再生単位にすることが望ましい。大まかな再生単位には、タイトル、PL といった非シームレスな再生単位が望ましい。一方、細かい再生単位としては、PL 内のチャプターというようにシームレスな再生単位が望ましい。アプリケーションの生存区間をタイトル毎、PL 毎に定めれば、アプリケーションはローカルメモリ 29 上に存在するので、そのタイトルの再生中においてアプリケーションは何時でも取り出せる状態になる。そうであれば、アプリケーションの生存区間を細かく定めたとしても、アプリケーションを即座に、仮想マシン上のワークメモリに読み出すことができるので、アプリケーションの起動・終了が頻繁になされたとしても、スムーズなアプリケーション実行を実現することができる。

次に、読込属性について説明する。

図 2 において Java アーカイブファイルは、AVClip とは別の記録領域に記録されることを前提にしていた。しかしこれは一例に過ぎない。Java アーカイブファイルは、BD-ROM において AVClip が占める記録領域に埋め込まれることがある。この埋め込みの態様には、カルーセル化、インターリーブユニット化という 2 種類がある。

ここで“カルーセル化”とは、対話的な放送の実現のために同一内容を繰り返すという放送方式に変換することである。BD-ROM は、放送されたデータを格納するものではないが、本実施形態では、カルーセルの放送形式に倣って JAVA アーカイブファイルを格納するようにしている。図 4 2 は、カルーセル化による Java アーカイブファイル埋め込みを示す図である。第 1 段目は、AVClip 中に埋め込む Java アーカイブファイルであり、第 2 段目は、セクション化を示す。第 3 段目は、TS パケット化、第 4 段目は、AVClip を構成する TS パケット列を示す。こうしてセクション化、TS パケット化されたデータ(図中の“D”)が、AVClip に埋め込まれるのである。カルーセルにより AVClip に多重化された Java アーカイブファイルは、読み出すにあたって、低帯域で読

み出されることになる。この低帯域での読み出しは、概して 2～3 分というように長期間を要するため、再生装置は Java アーカイブファイルを 2～3 分をかけて読み込むことになる。

図 4 3 は、インターリーブ化による Java アーカイブファイル埋め込みを示す図である。第 1 段目は、埋め込まれるべき AVClip、第 2 段目は、AVClip にインターリーブ化された Java アーカイブファイル、第 3 段目は、BD-ROM の記録領域における AVClip 配置である。本図に示すように、ストリームに埋め込まれるべき Java アーカイブファイルは、インターリーブ化され、AVClip を構成する XXXXX.m2ts を構成する分割部分(図中の AVClip2/4,3/4)の合間に記録される。インターリーブ化により AVClip に多重化された Java アーカイブファイルは、カルーセル化と比較して、高い帯域で読み出されることになる。この高い帯域での読み出しであるため、再生装置は Java アーカイブファイルを比較的短期間に読み込むことになる。

カルーセル化・インターリーブ化された Java アーカイブファイルは、プリロードされるのではない。BD-ROM における AVClip の記録領域のうち、カルーセル化・インターリーブ化された Java アーカイブファイルが埋め込まれた部分に、現在の再生時点が到達した際、再生装置のローカルメモリ 29 にロードされる。Java アーカイブファイルの記録態様には、図 2 に示すものの他に、図 4 2、図 4 3 (a) に示すものがあるので、読込属性は、図 4 3 (b) に示すように、設定されうる。図 4 3 (b) に示すように、読込属性は、タイトル再生に先立ち、ローカルメモリ 29 に読み込まれる旨を示す "Preload" と、タイトル再生中に、カルーセル化方式で読み込まれる旨を示す "Load.Carousel" と、タイトル再生中に、インターリーブ化方式で読み込まれる旨を示す "Load.InterLeave" とがある。読込属性には、カルーセル化されているか、インターリーブ化されているかが添え字で表現されているが、これを省略してもよい。

データ管理テーブルにおける生存区間の具体的な記述例について、図 4 4 を参照しながら説明する。図 4 4 (a) は、データ管理テーブルの一例を示す図である。図 4 4 (b) は、かかるデータ管理テーブルの割り当てによるローカ

ルメモリ 29 の格納内容の変遷を示す図である。本図は、縦軸方向にローカルメモリ 29 における占有領域を示し、横軸を、1 つのタイトル内の PL 時間軸としている。データ管理テーブルにおいて application#1 は、1 つのタイトル内の PL 時間軸全体を生存区間とするよう記述されているので、このタイトルの Chapter#1~Chapter#5 においてローカルメモリ 29 内の領域を占有することになる。データ管理テーブルにおいて application#2 は、タイトル内の PL#1 における Chapter#1~Chapter#2 を生存区間とするよう記述されているので、このタイトルの Chapter#1~Chapter#2 においてローカルメモリ 29 内の領域を占有することになる。データ管理テーブルにおいて application#3 は、タイトル内の PL#1 における Chapter#4~Chapter#5 を生存区間とするよう記述されているので、このタイトルの Chapter#4~Chapter#5 においてローカルメモリ 29 内の領域を占有することになる。以上で、データ管理テーブルにおける生存区間についての説明を終える。

続いて読込優先度について説明する。読込優先度とは、ローカルメモリ 29 への読み込みに対する優劣を決める優先度である。読込優先度には複数の値がある。2 段階の優劣を設けたい場合、Mandatory を示す値、optional を示す値を読込優先度に設定する。この場合、Mandatory は高い読込優先度を意味し、optional は、低い読込優先度を意味する。3 段階の優劣を設けたい場合、Mandatory を示す値、optional:high、optional:low を示す値を読込優先度に設定する。Mandatory は、最も高い読込優先度を示し、optional:high は、中程度の読込優先度、optional:low は、最も低い読込優先度を示す。データ管理テーブルにおける読込優先度の具体的な記述例について、図 45 (a) (b) を参照しながら説明する。この具体例で、想定しているローカルメモリ 29 のメモリ規模は、図 45 (a) に示すようなものである。図 45 (a) は、新旧再生装置におけるローカルメモリ 29 のメモリ規模を対比して示す図である。矢印 mk1 は旧再生装置におけるメモリ規模を、矢印 mk2 は新再生装置におけるメモリ規模をそれぞれ示す。この矢印の対比から、新再生装置におけるローカルメモリ 29 のメモリ規模は、旧再生装置のそれと比較して、三倍以上である状

態を想定している。このようにメモリ規模にバラツキがある場合、アプリケーションは、図45に示すような2つのグループに分類される。1つ目は、どのようなメモリ規模であっても読み込むんでおくべきアプリケーション(#1,#2)である。2つ目は、旧再生装置での読み込みは望まないが、新再生装置での読み込みは希望するアプリケーション(#3,#4)である。読み込もうとするアプリケーションが、これら2つのグループに分類されれば、前者に帰属するアプリケーションに、読込優先度=Mandatoryを設定し、後者に属するアプリケーションに、読込優先度=Optionalを設定する。図45(b)は、読込優先度が設定されたデータ管理テーブルの一例を示す図である。データ管理テーブルをこのように設定した上で、application#1~application#4をBD-ROMに記録すれば、あらゆるメモリ規模の再生装置での再生を保証しつつも、メモリ規模が大きい再生装置では、より大きなサイズのデータを利用したアプリケーションを再生装置に再生させることができる。

15 以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。上述した記録媒体の改良に対応するため、アプリケーションマネージャ36は図46に示すような処理手順で処理を行う。

図46は、アプリケーションマネージャ36によるプリロード制御の処理手順を示す図である。本フローチャートは、再生すべきタイトルにおけるデータ管理テーブルを読み込み(ステップS111)、データ管理テーブルにおいて最も高い読込優先度をもちつつ、applicationIDが最も小さいアプリケーションをアプリケーションiにした上で(ステップS112)、ステップS113、ステップS114の判定を経た上で、アプリケーションiをローカルメモリ29にプリロードする(ステップS115)という処理を、ステップS116がNo及び
25 ステップS117がNoと判定されるまで、繰り返すというループ処理を構成している。

ステップS113は、アプリケーションiの読込属性がプリロードであるか否かの判定であり、ステップS114は、アプリケーションの読込優先度が=MandatoryであるかOptionalであるかの判定である。ステップS113に

においてプリロードと判定され、ステップS 1 1 4において読込優先度がMandatoryと判定されれば、アプリケーションはローカルメモリ29にプリロードされることになる(ステップS 1 1 5)。もしステップS 1 1 3において読込属性がロードであると判定されれば、ステップS 1 1 4～ステップS 1 1 5はスキップされることになる。

ループ処理の終了要件を規定する2つのステップのうちステップS 1 1 6は、applicationIDが次に高く、アプリケーションiと同一読込優先度のアプリケーションkが存在するか否かを判定するものである。そのようなアプリケーションkが存在するなら、そのアプリケーションkをアプリケーションiにする(ステップS 1 1 9)。

ループ処理の終了要件を規定する2つのステップのうちステップS 1 1 7は、データ管理テーブルにおいて次に低い読込優先度をもつアプリケーションが存在するか否かの判定であり、もし存在すれば、その次に低い読込優先度をもつアプリケーションのうち、最も小さいapplicationIDをアプリケーションkを選んで(ステップS 1 1 8)、そのアプリケーションkをアプリケーションiにする(ステップS 1 1 9)。これらステップS 1 1 6、ステップS 1 1 7がYesになっている限り、上述したステップS 1 1 3～ステップS 1 1 5の処理は繰り返されることになる。ステップS 1 1 6、ステップS 1 1 7において、該当するアプリケーションが無くなれば本フローチャートの処理は終了することになる。

ステップS 1 2 0～ステップS 1 2 3は、ステップS 1 1 4において読込優先度=Optionalであると判定された場合に、実行される処理である。

ステップS 1 2 0は、同じapplicationIDをもち、読込優先度が高いアプリケーションjが存在するか否かの判定である。

ステップS 1 2 1は、ローカルメモリ29の残り容量がアプリケーションiのサイズを上回るか否かを判定するステップである。ステップS 1 2 0がNo、ステップS 1 2 1がYesである場合、ステップS 1 1 5においてアプリケーションiがローカルメモリ29にプリロードされることになる。ステップS 1 2 0がNo、ステップS 1 2 1がNoである場合、アプリケーションiはローカル

メモリ 29 にプリロードされずそのままステップ S 1 1 6 に移行することになる。

5 こうしておく、読み優先度=Optional のデータは、ステップ S 1 2 0 ～ステップ S 1 2 1 の判定が Yes にならないと、ローカルメモリ 29 へのプリロードがなされない。メモリ規模が小さい旧再生装置は、2～3 個のアプリケーションを読み込んだ程度で、ステップ S 1 2 1 の判定は No になるが、メモリ規模
10 が大きい新再生装置は、更に多くのアプリケーションを読み込んだとしても、ステップ S 1 2 1 の判定は No にならない。以上のように、旧再生装置では、ローカルメモリ 29 に Mandatory のアプリケーションのみが読み込まれ、新再生装置には、Mandatory のアプリケーションと、Optional のアプリケーションとが読み込まれることになる。

 ステップ S 1 2 2 は、ステップ S 1 2 0 において Yes と判定された場合に実行されるステップである。同じ applicationID をもち、読み優先度が高いアプリケーション j がローカルメモリ 29 上に存在する場合、ローカルメモリ 29
15 の残り容量と、アプリケーション j のサイズとの和が、アプリケーション i のサイズを上回るか否かを判定し(ステップ S 1 2 2)、もし上回れば、アプリケーション i を用いてローカルメモリ 29 上のアプリケーション j を上書きすることによりプリロードする(ステップ S 1 2 3)。下回る場合は、アプリケーション i はローカルメモリ 29 にプリロードされずそのままステップ S 1 1 6 に
20 移行することになる。

 ステップ S 1 1 5、ステップ S 1 2 3 による読み処理の一例を、図 4 7 (a) を参照しながら説明する。図 4 7 (a) は、この具体例が想定しているデータ管理テーブルの一例を示す図である。本図における 3 つのアプリケーションは、それぞれ 3 つのファイルに格納されており、applicationID は同じであるが
25 (applicationID=1)、読み優先度は互いに異なる (mandatory, optional:high, optional:low)。こうしたデータ管理テーブルが処理対象であると、ステップ S 1 1 5 により、読み優先度=Mandatory のアプリケーションはローカルメモリ 29 に読み込まれる。しかし読み優先度=Optional のアプリケーションについては、ステップ S 1 2 0 ～ステップ S 1 2 2 の判定

を経た上で、ステップ S 1 2 3 において読み込まれる。ステップ S 1 1 5 と
異なるステップ S 1 2 3 では、既にローカルメモリ 2 9 にある同じ applicationID
のアプリケーションを上書きしてゆくよう、プリロードがなされるので、複数
アプリケーションのうち 1 つが排他的に、ローカルメモリ 2 9 にロードされる
ことになる。

i) 読込優先度 = mandatory のアプリケーションを読み込んだ後、読込優先度
= optional:high のアプリケーションを読み込むにあたって、ステップ S 1 2 2
が No と判定されれば、読込優先度 = mandatory のアプリケーションがロー
カルメモリ 2 9 に残ることになる。読込優先度 = mandatory のアプリケーション
を読み込んだ後、読込優先度 = optional:high のアプリケーションを読み込む
にあたって、ステップ S 1 2 2 が Yes と判定されれば、読込優先度 =
optional:high のアプリケーションにより、読込優先度 = mandatory のアプリ
ケーションは上書きされ、読込優先度 = optional:high のアプリケーションがロー
カルメモリ 2 9 に残ることになる。

ii) 読込優先度 = optional:high のアプリケーションを読み込んだ後、読込優先
度 = optional:low のアプリケーションを読み込むにあたって、ステップ S 1 2
2 が No と判定されれば、読込優先度 = Mandatory のアプリケーションが
ローカルメモリ 2 9 に残ることになる。読込優先度 = optional:high のアプリ
ケーションを読み込んだ後、読込優先度 = optional:low のアプリケーションを
読み込むにあたって、ステップ S 1 2 2 が Yes と判定されれば、読込優先度 =
optional:low のアプリケーションにより、読込優先度 = optional:high のアプリ
ケーションは上書きされ(ステップ S 1 2 3)、読込優先度 = optional:low のア
pplication がローカルメモリ 2 9 に残ることになる。

ローカルメモリ 2 9 の容量が許す限り、ローカルメモリ 2 9 上のアプリケー
ションを上書きしてゆくとの処理が繰り返されるので、ローカルメモリ 2 9 の
格納内容は、図 4 7 (b) に示すように、mandatory = optional:high =>
optional:low と遷移してゆくことになる。メモリ規模に応じて、サイズが異な
る Java アーカイブファイルをローカルメモリ 2 9 にロードすることができる。

ので、メモリ規模が小さい再生装置については、必要最小限の解像度をもったサムネイル画像を有する Java アーカイブファイルを、メモリ規模が中程度の再生装置については、中程度の解像度をもった SD 画像を有する Java アーカイブファイルを、メモリ規模が大規模である再生装置については、高解像度をもった HD 画像を有する Java アーカイブファイルをローカルメモリ 29 にロードすることができる。かかるロードにより、メモリ規模に応じて解像度が異なる画像を表示させることができ、オーサリング担当者によるタイトル制作の表現の幅が広がる。

- 10 図 48 は、データ管理テーブルを参照した読取処理の具体例を示す図である。本図における 2 つのアプリケーションは、同じ applicationID(application#3) が付与された 2 つのアプリケーションを示す図である。そのうち一方は、AVClip 中に埋め込まれていて、読込優先度が mandatory に設定されている。他方は、AVClip とは別ファイルに記録されていて、読込優先度が Optional に
- 15 設定されている。前者のアプリケーションは、AVClip に埋め込まれているので、その埋込部分にあたる生存区間が、生存区間(title#1:chapter#4~#5)として記述されている。これらのアプリケーションのうち application#2、application#3 には、ロードを示す読込属性が付与されている。application#2 は Chapter#1~Chapter#2 を生存区間にしており、application#3 は Chapter#4
- 20 ~Chapter#5 を生存区間にしていて、タイトル時間軸においてどちらか一方が排他的にローカルメモリ 29 上に常駐することになる。図 48 (b) は、タイトル時間軸上の別々の時点において、排他的に格納される application#2、application#3 を示す図である。これは必要最低限のメモリ規模しかもたない再生装置での再生を念頭に置いた配慮である。こうした内容のデータ管理テーブルが処理対象であるとアプリケーションマネージャ 36 は、上述した図 46
- 25 のフローチャートによりメモリ規模に応じて異なる処理を行う。

後者のアプリケーションは、読込優先度=ロードであるので、ローカルメモリ 29 にロードされる。かかる処理により、Mandatory なメモリ規模さえあれば、アプリケーションマネージャはデータをローカルメモリ 29 にロードする

ことができる。ここで問題になるのは、メモリ規模が大きい再生装置による読み込み時である。メモリ規模が大きいにも拘らず、Chapter#4～Chapter#5に到達するまで application#3 を読み込めないというのは、メモリ規模の無駄になる。そこで本図のデータ管理テーブルには、同じ application#3 にプリロードを示す読込属性を付与して BD-ROM に記録しておき、これらに同じ applicationID を付与している。

前者のアプリケーションは、読込優先度=Optional であるので、ステップ S 1 2 1 が Yes になった場合に限り、プリロードされる(ステップ S 1 1 5)。こうすることで、メモリ規模が大きい再生装置は、title#1、Chapter#4～Chapter#5 の到達を待つことなく、AVClip に埋め込まれているのと同じアプリケーションをローカルメモリ 2 9 にロードすることができるのである(図 4 8 (c))。

以上がプリロード時における処理である。続いてロード時における処理手順について説明する。

図 4 9 は、データ管理テーブルに基づくロード処理の処理手順を示す図である。本フローチャートは、ステップ S 1 3 1～ステップ S 1 3 3 からなるループ処理を、タイトル再生が継続されている間、繰り返すというものである。

ステップ S 1 3 1 は、AutoRun を示す起動属性を有したアプリケーションの生存区間が到来したか否かの判定である。もし到来すれば、AutoRun を示す起動属性を有したアプリケーションをアプリケーション q にして(ステップ S 1 3 4)、アプリケーション q を起動する旨の起動指示を Java 仮想マシン 3 8 に発行して、アプリケーション q をローカルメモリ 2 9 からワークメモリ 3 7 に読み出させる(ステップ S 1 3 5)。

ステップ S 1 3 3 は、タイトル内 PL の再生が全て終了したかの判定である。この判定は、第 5 実施形態に示したように、Playback Control Engine 3 2 からの再生終結イベントがあったか否かでなされる。もし終了すれば、本フローチャートの処理を終了する。

ステップ S 1 3 2 は、起動中アプリケーションからの呼出があったか否かの判定である。もしあれば、呼出先アプリケーションをアプリケーション q にし

て(ステップS 1 3 6)、現在の再生時点は、アプリケーション管理テーブルにおけるアプリケーション q の生存区間であるか否かを判定する(ステップS 1 3 7)。もし生存区間でなければ、起動失敗を表示して(ステップS 1 4 8)、ステップS 1 3 1～ステップS 1 3 3からなるループ処理に戻る。生存区間であれば、図50のフローチャートに従い、ロード処理を行う。

図50におけるステップS 1 3 8は、現在の再生時点がデータ管理テーブルにおけるアプリケーション q の生存区間であるか否かを示す判定である。もし生存区間でなければ、アプリケーション q はローカルメモリ29にロードすることができない。この場合、アプリケーション q を起動する旨の起動指示をJava 仮想マシン38に発行し、ローカルメモリ29を介することなく、直接アプリケーション q をBD-ROM からワークメモリ37に読み出させる。この場合アプリケーションを読み出すためのヘッドシークが発生するから、PL 再生は中断することになる(ステップS 1 4 5)。

もし生存区間であれば、ステップS 1 3 9において、アプリケーションには読込属性が付加されているか否かを判定する。読込属性がないということは、アプリケーション q は、カルーセル化、若しくはインターリーブ化されていないことを意味する。しかし読込属性が付加されていなくても、ローカルメモリ29にアプリケーション q を置くことは許される。そこで再生中断を承知の上、アプリケーションの読み出しを行う。つまり BD-ROM からローカルメモリ29へとアプリケーションを読み出した上で、アプリケーションをワークメモリ37に読み出す(ステップS 1 4 0)。

ステップS 1 4 1～ステップS 1 4 6は、ステップS 1 3 9がYesと判定された場合になされる処理である。ステップS 1 4 1では、読込属性を参照することで、アプリケーションがプリロードされているか否かを判定する。プリロードされていれば、ステップS 1 3 5に移行する。

ステップS 1 4 2は、読込属性がロードである場合に実行される判定ステップであり、アプリケーション q がカルーセル化されているか、インターリーブ化されているかを判定する。インターリーブ化されていれば、キャッシュセンスをJava 仮想マシン38に実行させる(ステップS 1 4 3)。ローカルメモリ2

9にアプリケーションqが存在すれば、ステップS135に移行して、アプリケーションqをJava仮想マシン38にロードさせる。

ローカルメモリ29にアプリケーションがなければ、トップメニュータイトルに分岐する等の例外処理を行う(ステップS144)。カールセル化されてい
5 れば、タイマをセットし(ステップS148)、そのタイマがタイムアウトする
まで(ステップS147)、キャッシュセンスをJava仮想マシン38に実行させる(ステップS146)。もしローカルメモリ29にアプリケーションqが出現
すれば、図49のステップS135に移行して、アプリケーションqをJava
仮想マシン38にロードさせる。タイムアウトすれば、トップメニュータイト
10 ルに分岐する等の例外処理を行う(ステップS144)。

図51は、Java仮想マシン38によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

矢印◎1,2は、アプリケーション管理テーブルに生存していて、データ管理
テーブルに生存しており、カールセル化、インターリーブ化を示す読込属性が存
15 在するJavaアーカイブファイルの読み込みを示す。矢印◎1は、ステップS
65、67においてなされるローカルメモリ29センスを示す。このローカル
メモリ29センスは、カールセル又はインターリーブ化により埋め込まれたデ
ータが、ローカルメモリ29に存在するかもしれないためローカルメモリ29
内をセンスするというものである。矢印◎2は、ステップS135に対応する
20 読み込みであり、アプリケーションがローカルメモリ29に存在していた場合
の、ローカルメモリ29からワークメモリ37へのロードを示す。×付きの矢
印は、ローカルメモリ29にデータがない場合を示す。

矢印▽1,2は、アプリケーション管理テーブルに生存しているが、データ管
理テーブルに生存しておらず、読込属性が存在しないJavaアーカイブファイ
25 ルの読み込みを示す。

矢印▽1は、ステップS145における読み込みに対応するものであり、Java
仮想マシン38によるBD-ROMからのダイレクトリードの要求を示す。矢印
▽2はその要求による、BD-ROMからワークメモリ37へのJavaアーカイブ
ファイル読み出しを示す。

矢印☆1,2,3 は、アプリケーション管理テーブルに生存していて、データ管理テーブルに生存しているが、読込属性が存在しない Java アーカイブファイルの読み込みを示す。

- 矢印☆1 は、ステップ S 1 4 0 における読み込みに対応するものであり、Java
5 仮想マシン 3 8 による BD-ROM からのダイレクトリードの要求を示す。矢印
☆2 はその要求による、ローカルメモリ 2 9 への Java アーカイブファイルの
読み出しを示す。矢印☆3 はローカルメモリ 2 9 からワークメモリ 3 7 への
Java アーカイブファイルの読み出しを示す。

- 以上のように本実施形態によれば、ローカルメモリ 2 9 上で同時に常駐され
10 るアプリケーションの数が所定数以下になるように規定しておくことができる
ので、ローカルメモリ 2 9 からの読み出し時におけるキャッシュミスを極力回
避することができる。キャッシュミスのないアプリケーション読み出しを保証
することができるので、アプリケーション呼出時にあては、AVClip の再生
を止めてまで、BD-ROM からアプリケーションを読み出すことはなくなる。
15 AVClip 再生を途切れさせないので、AVClip のシームレス再生を保証すること
ができる。

(第 7 実施形態)

- 第 3 実施形態では、非 AV 系タイトルの時間軸をアプリケーションの生存区
間に基づき定めることにした。しかしアプリケーションの動作というのは不安
20 定であり、起動の失敗や異常終了がありうる。本実施形態は、起動失敗、異常
終了があった場合の Fail Safe 機構を提案するものである。図 5 2 (a) は、
第 7 実施形態に係る BD-J オブジェクトの内部構成を示す図である。図 7 (b)
と比較して本図が新規なのは、プレイリスト管理テーブルが追加されている点
である。

- 25 図 5 2 (b) は、プレイリスト管理テーブルの一例を示す図である。本図に
示すようにプレイリスト管理テーブルは、PL の指定と、その PL の再生属性と
からなる。PL の指定は、対応するタイトルのタイトル時間軸において、再生
可能となる PL を示す。PL の再生属性は、指定された PL を、タイトル再生の
開始と同時に自動再生するか否かを示す(こうして自動再生される PL をデフォ

ルト PL という)。

次にプレイリスト管理テーブルによりタイトル時間軸がどのように規定されるかを、図 5 3 を参照しながら説明する。図 5 3 (a) は、再生属性が非自動再生を示すよう設定された場合の非 AV 系タイトルにおけるタイトル時間軸を示す図である。この場合、デフォルト PL は再生されないから、非 AV 系タイトル同様、アプリケーションの生存区間からタイトル時間軸が定まる。

図 5 3 (b) は、再生属性が AutoPlay に設定された非 AV 系タイトルのタイトル時間軸を示す図である。再生属性が AutoPlay を示すよう設定されれば、Playback Control Engine 3 2 は非 AV 系タイトルの再生開始と同時に、デフォルト PL の再生を開始する。しかしアプリケーションが正常に動作し、正常終了したとしても、このタイトル時間軸は、PL 時間軸を基準にして定められる。

図 5 3 (c) は、プレイリスト管理テーブルにおいて再生属性が "AutoPlay" を示すよう設定され、アプリケーションが異常終了した場合を示す。かかる異常終了により、どのアプリケーションも動作してない状態になるが、デフォルト PL の再生は継続する。この場合も、デフォルト PL の PL 時間軸がタイトル時間軸になる。

図 5 3 (d) は、プレイリスト管理テーブルにおいて再生属性が "AutoPlay" を示すよう設定され、メインアプリの起動に失敗したケースを示す。この場合も、Playback Control Engine 3 2 によるデフォルト PL 再生は、アプリケーションの起動失敗とは関係なしに行われるので、デフォルト PL の時間軸がタイトル時間軸になる。

以上のようにプレイリスト管理テーブルの再生属性を、"AutoPlay" に設定しておけば、Java アプリケーションの起動に、5~10 秒という時間がかかったとしても、その起動がなされている間、"とりあえず何かが写っている状態" になる。この "とりあえず何かが写っている状態" によりタイトル実行開始時のスタートアップディレイを補うことができる。

以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。

図 5 2 (c) は、分岐先タイトルのプレイリスト管理テーブルにおいて、再

生属性が AutoPlay に設定された PL が存在する場合、再生装置がどのような処理を行うかを示す図である。本図に示すように、再生属性が AutoPlay に設定された PL が、分岐先タイトルのプレイリスト管理テーブルに存在すれば、BD-J モジュール 35 内のアプリケーションマネージャ 36 は、タイトル分岐直後にこの AutoPlay PL の再生を開始するよう Playback Control Engine 32 に指示する。このように再生属性が AutoPlay の PL は、タイトル分岐直後に再生開始が命じられることになる。

上述した記録媒体の改良に対応するため、アプリケーションマネージャ 36 は図 54 に示すような処理手順で処理を行う。

- 10 図 54 は、第 7 実施形態に係るアプリケーションマネージャ 36 の処理手順を示すフローチャートである。本フローチャートは、図 38 のフローチャートにおいてステップ S 21 の前にステップ S 103、ステップ S 104 を追加し、ステップ S 21 と、ステップ S 22 との間にステップ S 100 を追加し、ステップ S 23ーステップ S 26 間に、ステップ S 105 を追加したものである。
- 15 ステップ S 103 は、対応するタイトルのプレイリスト管理テーブルの再生属性が AutoPlay であるか否かの判定である。もし AutoPlay なら、デフォルト PL に対する再生制御を Playback Control Engine 32 に開始させる(ステップ S 104)。

- ステップ S 100 は、Presentation Engine 31 による再生中であるか否かを判定する。もし再生中であるなら、ステップ S 101 に移行する。
- 20

- ステップ S 105 は、ステップ S 23 が Yes、ステップ S 25 が No である場合に実行される判定ステップであり、再生属性が AutoPlay であるか否かを示す。もし否であるなら、タイトル終了をモジュールマネージャ 34 に通知する。もし AutoPlay であるなら、ステップ S 101 に移行して、処理を継続する。
- 25

図 55 は、プレイリスト管理テーブルにおいて”再生属性=AutoPlay”に設定されることにより、どのような再生が行われるかを模式化した図である。ここで再生すべきタイトルは、落下するタイル片を積み重ねるというゲームアプリを含む非 AV 系タイトルである。この非 AV 系タイトルにおいて、プレイリ

スト管理テーブルの再生属性が AutoPlay に設定されていれば、Playback Control Engine 3 2 によるデフォルト PL 再生も開始する。ゲームアプリの実行と、デフォルト PL 再生とが並列的になされるので、図 5 5 の上段の左側に示すように、前景をゲームアプリの画面とし、背景をデフォルト PL の再生画像とした合成画像が表示されることになる。このゲームアプリは途中で異常終了したとする。ゲームアプリはアプリケーションマネージャ 3 6 により強制終了させられるが、デフォルト PL の再生が継続してなされるため、タイトルは、何かが写っている状態になる。このようなプレイリスト管理テーブルにおける再生属性の指定により、非 AV 系タイトル内のゲームアプリが異常終了した場合でも、ハングアップやブラックアウトがない動作を維持することができる。

(第 8 実施形態)

第 1 実施形態において BD-J オブジェクトは、データ管理テーブル、アプリケーション管理テーブルという 2 つのテーブルを具備していたが、本実施形態は、これらを 1 つのテーブルに統合するという形態を開示する。かかる統合にあたって、図 5 6 (a) に示すように、データ管理テーブルにおける読込属性という項目を廃し、代わりに起動属性に Ready 属性という属性を設ける。Ready 属性とは、他のアプリケーションからの呼出又はアプリケーションマネージャ 3 6 からの呼出に備えて、ローカルメモリ 2 9 に予めアプリケーションをロードしておく旨を示す起動属性の類型である。

図 5 6 (b) は、アプリケーションの扱いと、起動属性との関係を示した図である。第 1 実施形態に示したようにアプリケーションの扱いには、プリロードされるか否か(1)、現在の再生時点が有効区間に到来した際自動的に起動されるか、他からの呼出に応じて起動されるか(2)、タイトル再生進行に従ってロードされるか(3)、生存しているかという違いがあり、これらの違いにより、図 5 6 (b) に示すような 5 つの態様が出現する。このうち起動属性が AutoRun に設定されるのは、プリロードがなされ、“自動起動”である場合、及び、ロードがなされ、“自動起動”である場合である。

一方、起動属性が Ready 属性に設定されるのは、プリロード、又は、ロードがなされ、起動項目が“呼出起動”を示している場合である。

尚、ワークメモリ 37 では生存しているが、ローカルメモリ 29 にはロードされない”との類型が存在し得ない。これは、アプリケーション・データ管理テーブルでは、ワークメモリ 37 の生存区間と、ローカルメモリ 29 の生存区間とが一体だからである。

- 5 起動属性として、この Ready 属性を追加されたので、アプリケーションマネージャ 36 はタイトル再生に先立ち、起動属性が AutoRun に設定されたアプリケーション、及び、起動属性が Ready 属性に設定されたアプリケーションをローカルメモリ 29 にプリロードするとの処理を行う。こうすることにより、読込属性を設けなくても、アプリケーションをローカルメモリ 29 にプリロードしておくとの処理が可能になる。
- 10 ドしておくとの処理が可能になる。

図 57 は、第 8 実施形態に係る Java 仮想マシン 38 によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。本図における読み込みは、図 51 をベースにして作図している。

- 矢印◎1,2 は、アプリケーション・データ管理テーブルに生存していて、起動属性が Ready 属性に設定されている Java アーカイブファイルの読み込みを示す。
- 15 属性が Ready 属性に設定されている Java アーカイブファイルの読み込みを示す。

矢印☆1,2,3 は、アプリケーション・データ管理テーブルに生存していおり、起動属性が Persistent であるアプリケーションの読み込みを示す。

- これらの矢印◎1,2、矢印☆1,2,3 は、図 51 でも記述されていたものだが、図 51 に記述していた、▽1,2 の矢印に該当する読み込み”は、図 57 では存在しない。これは、アプリケーション・データ管理テーブルは、アプリケーション管理テーブル、データ管理テーブルを一体化したものであるため、アプリケーション管理テーブル=生存、データ管理テーブル=非存在という組合せは表現し得ないからである。
- 20 図 51 に記述していた、▽1,2 の矢印に該当する読み込み”は、図 57 では存在しない。これは、アプリケーション・データ管理テーブルは、アプリケーション管理テーブル、データ管理テーブルを一体化したものであるため、アプリケーション管理テーブル=生存、データ管理テーブル=非存在という組合せは表現し得ないからである。

- 25 以上のように本実施形態によれば、データ管理テーブル、アプリケーション管理テーブルを 1 つのテーブル(アプリケーション・データ管理テーブル)にまとめることができるので、アプリケーションマネージャ 36 による処理を簡略化することができる。尚、読込優先度をなくすことによりアプリケーション・データ管理テーブルをより簡略化にしても良い。

(第9実施形態)

第1実施形態では、アプリケーションをローカルメモリ29に読み込むにあたって、読込優先度を参照して、この読込優先度に従い、読み込み処理に優劣を与えた。これに対し第9実施形態は、Optionalを意味する情報と、0から255
5 までの数値との組合せにより読込優先度を表す実施形態である。

図58(a)(b)は、第9実施形態に係る読込優先度の一例を示す図である。255、128は、0から255までの読込優先度の一例であり、本例におけるapplication#2は、application#3より読込優先度が高いことを意味する。

本実施形態においてアプリケーションマネージャ36は、第1実施形態同様、
10 先ずMandatoryを示す読込優先度が付与されたアプリケーションをローカルメモリ29に読み込む。

その後、Optionalを示す読込優先度が付与されたアプリケーションに対しては、ローカルメモリ29における容量が、アプリケーションのサイズを上回るか否かを判定する。もし上回るなら、読込優先度=Optionalが付与されたアプリケーションをそのままローカルメモリ29に読み込む。もし下回るなら、アプリケーションを構成するデータのうち、読込優先度を表す数値が高いアプリケーションをローカルメモリ29に読み込む。そして、ローカルメモリ29における残りの領域に、読込優先度を表す数値が低いアプリケーションを読み出す。
15

20

こうすることでOptional扱いのアプリケーションについては、全体を格納する容量が再生装置のローカルメモリ29になくても、その一部分をローカルメモリ29に格納しておくことができる。

(第10実施形態)

25 第1実施形態においてアプリケーションマネージャ36は、同じapplicationIDが付与されたアプリケーションを、読込優先度に従い排他的にローカルメモリ29にロードするとしたが、第10実施形態は、アプリケーションにグループ属性を与えることにより、排他的なロードを実現する。図59は、グループ属性が付与されたデータ管理テーブルを示す図である。グループ

属性には、排他グループなし、排他グループあり、といった、2通りの設定が可能であり、排他グループありの場合、そのグループ番号が記述される。図59(a)におけるtitle#1の「-」は、排他グループが存在しないことを示す。一方、title#2,#3の「group#1」は、排他グループがあり、title#2,#3は、group#1
5 という排他グループに帰属していることを示す。以上が本実施形態に係る記録媒体の改良である。

本実施形態に係る再生装置は、データ管理テーブルに基づいて各アプリケーションをローカルメモリ29に読み込んだ後、ローカルメモリ29のアプリケーションにおけるグループ属性をベリファイする。同じ排他グループに帰属するアプリケーションが、ローカルメモリ29上に2つ以上存在していれば、そのうち一方をローカルメモリ29から削除する。
10

こうすることにより、ローカルメモリ29の利用効率を向上させることができる。排他グループの具体例としては、ランチャーアプリと、このアプリにより起動されるアプリとからなるグループが相応しい。本アプリケーションにより
15 起動されるアプリケーションは、原則1つに限られるので、ローカルメモリ29には、ランチャー+1個のアプリケーションのみが存在する筈である。もし3つ以上のアプリケーションが存在していれば、これをローカルメモリ29から削除するという処理をアプリケーションマネージャ36は行う必要がある
20 ので、各アプリケーションのグループ属性を設け、ローカルメモリ29上で存在するアプリケーションがランチャー+1個のアプリケーションになっているかどうかのチェックを行うのである。

図59(a)は、アプリケーション管理テーブルに基づくローカルメモリ29に対するアクセスを示す図である。本図において、読込優先度=Optionalと設定されたapplication#2、application#3のグループ属性は、group#1である
25 ので、これらのアプリケーションは、同じ排他グループに属することになる。3つのアプリケーションのうち、application#1は上述したランチャーアプリケーションであり、application#2、application#3は、これにより起動されるアプリケーションであるので、どちらかのみがローカルメモリ29上に存在するよう、グループ属性が付与されている。アプリケーションマネージャ36は、

これら application#2、application#3 のグループ属性を参照して、どちらか 1 つをローカルメモリ 29 から削除するとの処理を行う。かかる削除によりローカルメモリ 29 に余白が生まれる。

(第 1 1 実施形態)

- 5 第 1 実施形態では、アプリケーション管理テーブルをタイトル毎に持たせるとしたが、本実施形態では、このアプリケーション管理テーブルの割当単位を変更させることを提案する。図 60 は、割当単位のバリエーションを示す図である。本図において第 1 段目は、BD-ROM に記録されている 3 つのアプリケーション管理テーブルを示し、第 2 段目は、タイトル単位、第 3 段目は、ディスク単位、第 4 段目は、複数 BD-ROM からなるディスクセット単位を示す。図中の矢印は、アプリケーション管理テーブルの割り当てを模式化して示している。この矢印を参照すると、第 1 段目におけるアプリケーション管理テーブル#1,#2,#3 のそれぞれは、第 2 段目に示した title#1,#2,#3 のそれぞれに割り当てられていることがわかる。また、ディスク単位ではアプリケーション管理テーブル#4 が割り当てられており、ディスクセット全体に対してはアプリケーション管理テーブル#5 が割り当てられている。このようにアプリケーション管理テーブルの割当単位を、タイトルより大きい単位にすることにより、1 つの BD-ROM がローディングされている間、生存するようなアプリケーションや複数 BD-ROM のうちどれかがローディングされている間、生存するようなアプリケーションを定義することができる。
- 10
15
20

(備考)

- 以上の説明は、本発明の全ての実施行為の形態を示している訳ではない。下記(A)(B)(C)(D)……の変更を施した実施行為の形態によっても、本発明の実施は可能となる。本願の請求項に係る各発明は、以上に記載した複数の実施形態及びそれらの変形形態を拡張した記載、ないし、一般化した記載としている。拡張ないし一般化の程度は、本発明の技術分野の、出願当時の技術水準の特性に基づく。
- 25

(A)全ての実施形態では、本発明に係る光ディスクを BD-ROM として実施し

たが、本発明の光ディスクは、記録される動的シナリオ、Index Table に特徴があり、この特徴は、BD-ROM の物理的性質に依存するものではない。動的シナリオ、Index Table を記録しうる記録媒体なら、どのような記録媒体であってもよい。例えば、

- 5 DVD-ROM,DVD-RAM,DVD-RW,DVD-R,DVD+RW,DVD+R,CD-R,CD-RW 等の光ディスク、PD,MO 等の光磁気ディスクであってもよい。また、コンパクトフラッシュカード、スマートメディア、メモリスティック、マルチメディアカード、PCM-CIA カード等の半導体メモリカードであってもよい。フレキシブルディスク、SuperDisk,Zip,Click! 等の磁気記録ディスク (i)、
- 10 ORB,Jaz,SparQ,SyJet,EZFley,マイクロドライブ等のリムーバルハードディスクドライブ(ii)であってもよい。更に、機器内蔵型のハードディスクであってもよい。

- (B) 全ての実施形態における再生装置は、BD-ROM に記録された AVClip を
- 15 デコードした上で TV に出力していたが、再生装置を BD-ROM ドライブのみとし、これ以外の構成要素を TV に具備させてもよい、この場合、再生装置と、TV とを IEEE1394 で接続されたホームネットワークに組み入れることができる。また、実施形態における再生装置は、テレビと接続して利用されるタイプであったが、ディスプレイと一体型となった再生装置であってもよい。更に、
- 20 各実施形態の再生装置において、処理の本質的部分をなす部分のみを、再生装置としてもよい。これらの再生装置は、何れも本願明細書に記載された発明であるから、これらの何れの態様であろうとも、各実施形態に示した再生装置の内部構成を元に、再生装置を製造する行為は、本願の明細書に記載された発明の実施行為になる。各実施形態に示した再生装置の有償・無償による譲渡(有償
- 25 の場合は販売、無償の場合は贈与になる)、貸与、輸入する行為も、本発明の実施行為である。店頭展示、カタログ勧誘、パンフレット配布により、これらの譲渡や貸渡を、一般ユーザに申し出る行為も本再生装置の実施行為である。

(C)各フローチャートに示したプログラムによる情報処理は、ハードウェア資源を用いて具体的に実現されていることから、上記フローチャートに処理手順

を示したプログラムは、単体で発明として成立する。全ての実施形態は、再生装置に組み込まれた態様で、本発明に係るプログラムの実施行為についての実施形態を示したが、再生装置から分離して、各実施形態に示したプログラム単体を実施してもよい。プログラム単体の実施行為には、これらのプログラムを生産する行為(1)や、有償・無償によりプログラムを譲渡する行為(2)、貸与する行為(3)、輸入する行為(4)、双方向の電子通信回線を介して公衆に提供する行為(5)、店頭展示、カタログ勧誘、パンフレット配布により、プログラムの譲渡や貸渡を、一般ユーザに申し出る行為(6)がある。

(D)各フローチャートにおいて時系列に実行される各ステップの「時」の要素を、発明を特定するための必須の事項と考える。そうすると、これらのフローチャートによる処理手順は、再生方法の使用形態を開示していることがわかる。各ステップの処理を、時系列に行うことで、本発明の本来の目的を達成し、作用及び効果を奏するよう、これらのフローチャートの処理を行うのであれば、本発明に係る記録方法の実施行為に該当することはいうまでもない。

(E)Chapterを一覧表示するためのMenu(Chapter Menu)と、この挙動を制御するMOVIEオブジェクトとをBD-ROMに記録しておき、Top Menuから分岐できるようにしてもよい。またリモコンキーのChapterキーの押下により呼出されるようにしてもよい。

(F)BD-ROMに記録するにあたって、AVClipを構成する各TSパケットには、拡張ヘッダを付与しておくことが望ましい。拡張ヘッダは、TP_extra_headerと呼ばれ、『Arrival_Time_Stamp』と、『copy_permission_indicator』とを含み4バイトのデータ長を有する。TP_extra_header付きTSパケット(以下EX付きTSパケットと略す)は、32個毎にグループ化されて、3つのセクタに書き込まれる。32個のEX付きTSパケットからなるグループは、6144バイト(=32×192)であり、これは3個のセクタサイズ6144バイト(=2048×3)と一致する。3個のセクタに収められた32個のEX付きTSパケットを”Aligned Unit”という。

IEEE1394を介して接続されたホームネットワークでの利用時において、再生装置200は、以下のような送信処理にてAligned Unitの送信を行う。つま

り送り手側の機器は、Aligned Unit に含まれる 32 個の EX 付き TS パケットのそれぞれから TP_extra_header を取り外し、TS パケット本体を DTCP 規格に基づき暗号化して出力する。TS パケットの出力にあたっては、TS パケット間の随所に、isochronous パケットを挿入する。この挿入箇所は、

5 TP_extra_header の Arribval_Time_Stamp に示される時刻に基づいた位置である。TS パケットの出力に伴い、再生装置 200 は DTCP_Descriptor を出力する。DTCP_Descriptor は、TP_extra_header におけるコピー許否設定を示す。ここで「コピー禁止」を示すよう DTCP_Descriptor を記述しておけば、IEEE1394 を介して接続されたホームネットワークでの利用時において TS

10 パケットは、他の機器に記録されることはない。

(G)各実施形態において、記録媒体に記録されるデジタルストリームは AVClip であったが、DVD-Video 規格、DVD-Video Recording 規格の VOB(Video Object)であってもよい。VOB は、ビデオストリーム、オーディオストリームを多重化することにより得られた ISO/IEC13818-1 規格準拠のプログラムストリームである。また AVClip におけるビデオストリームは、MPEG4

15 や WMV 方式であってもよい。更にオーディオストリームは、Linear-PCM 方式、Dolby-AC3 方式、MP3 方式、MPEG-AAC 方式、Dts、WMA(Windows media audio)であってもよい。

(H)各実施形態における映像作品は、アナログ放送で放送されたアナログ映像

20 信号をエンコードすることにより得られたものでもよい。デジタル放送で放送されたトランスポートストリームから構成されるストリームデータであってもよい。

またビデオテープに記録されているアナログ／デジタルの映像信号をエンコードしてコンテンツを得ても良い。更にビデオカメラから直接取り込んだアナ

25 ログ／デジタルの映像信号をエンコードしてコンテンツを得ても良い。他にも、配信サーバにより配信されるデジタル著作物でもよい。

(I)BD-J モジュール 35 は、衛星放送受信のために機器に組み込まれた Java プラットフォームであってもよい。BD-J モジュール 35 がかかる Java プラットフォームであれば、本発明に係る再生装置は、MHP 用 STB としての処理を

兼用することになる。

- 更に携帯電話の処理制御のために機器に組み込まれた Java プラットフォームであってもよい。かかる BD-J モジュール 35 がかかる Java プラットフォームであれば、本発明に係る再生装置は、携帯電話としての処理を兼用することになる。

- (K)レイアモデルにおいて、BD-J モードの上に MOVIE モードを配置してもよい。特に MOVIE モードでの動的シナリオの解釈や、動的シナリオに基づく制御手順の実行は、再生装置に対する負担が軽いので、MOVIE モードを BD-J モード上で実行させても何等问题は生じないからである。また再生装置や映画作品の開発にあたって、動作保証が 1 つのモードで済むからである。

更に BD-J モードだけで再生処理を実行してもよい。第 5 実施形態に示したように、BD-J モードでも PL の再生と同期した再生制御が可能になるから、強いて MOVIE モードを設けなくてもよいという理由による。

- (L)AVClip に多重化されるべきインタラクティブグラフィクスストリームにナビゲーションコマンドを設けて、ある PL から別の PL への分岐を実現しても良い。

産業上の利用可能性

- 本発明に係る再生装置は、ホームシアターシステムでの利用のように、個人的な用途で利用されることがありうる。しかし本発明は上記実施形態に内部構成が開示されており、この内部構成に基づき量産することが明らかなので、資質において工業上利用することができる。このことから本発明に係る再生装置は、産業上の利用可能性を有する。

請求の範囲

1. タイトルの再生と、アプリケーションの実行とを行う再生装置であって、
タイトルに帰属するデジタルストリームを再生する再生制御エンジン部と、
5 複数タイトル間の分岐を制御するモジュールマネージャと、
1つ以上のアプリケーションを実行するモジュールとを備え、
前記モジュールは、仮想マシン部と、アプリケーションマネージャを含み、
前記アプリケーションマネージャは、
1つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了
10 したとして解釈して終了処理を行い、
前記モジュールマネージャは、前記タイトルの終了後、所定のタイトルを選択
する、ことを特徴とする再生装置。

2. アプリケーションにおける所定の状態とは、
15 1つ以上のアプリケーションが全て終了した状態である
ことを特徴とする請求項1記載の再生装置。

3. 1つ以上のアプリケーションには、メインアプリケーションがあり、
メインアプリケーションは、タイトル開始と同時に自動的に起動される唯一
20 のアプリケーションであり、
アプリケーションにおける所定の状態とは、
アプリケーションのうち、メインアプリケーションが終了した状態である、
請求項1記載の再生装置。

- 25 4. モジュールマネージャにより選択される所定のタイトルとは、トップメ
ニューの表示制御を実行するトップメニュータイトルである、請求項2又は3
記載の再生装置。

5. 前記記録媒体には、タイトル開始時において再生を自動的に開始すべき

再生経路を規定する規定情報が記述されており、

前記モジュールは、タイトル開始時において、アプリケーションの起動を仮想マシンに指示すると共に、前記規定情報に示される再生経路に従い、デジタルストリームの再生を開始するよう再生制御エンジン部に指示し、

5 再生制御エンジン部は、

仮想マシンによるアプリケーション実行が終了した際、所定の再生経路に従ったデジタルストリームの再生出力を継続して行う、請求項 1 記載の再生装置。

6. タイトルの再生と、アプリケーションの実行とを同時にコンピュータに
10 実行させるプログラムであって、

1 つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈してアプリケーションの終了処理を行い、前記タイトルの終了後、所定のタイトルを選択するようコンピュータを制御する、ことを特徴とするプログラム。

15

7. タイトルの再生と、アプリケーションの実行とを同時にコンピュータに
実行させる再生方法であって、

1 つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈してアプリケーションの終了処理を行い、前記タイトルの終了
20 後、所定のタイトルを選択するようコンピュータを制御する、ことを特徴とする再生方法。

THIS PAGE BLANK (USPTO)

図1

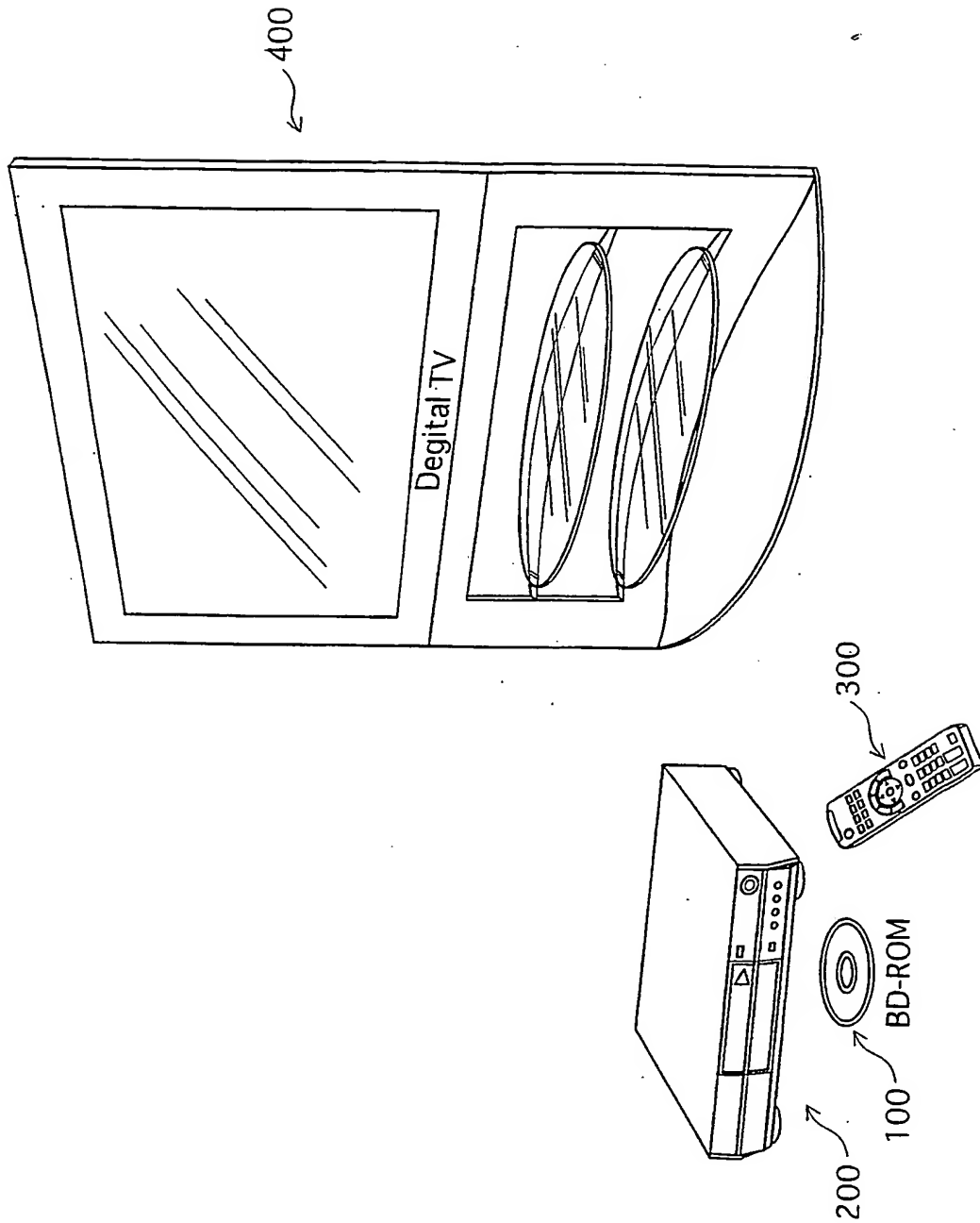


図2

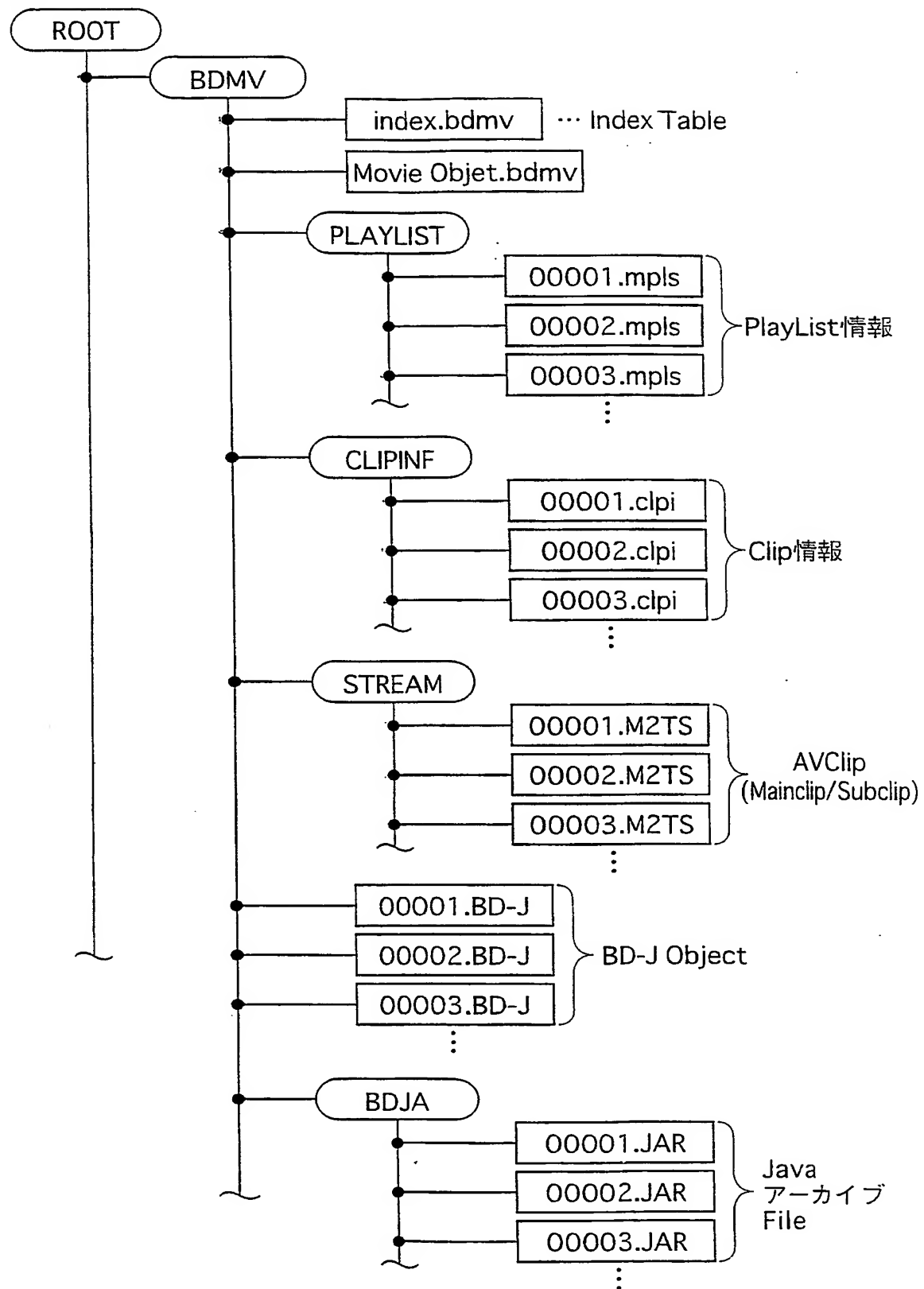


図3

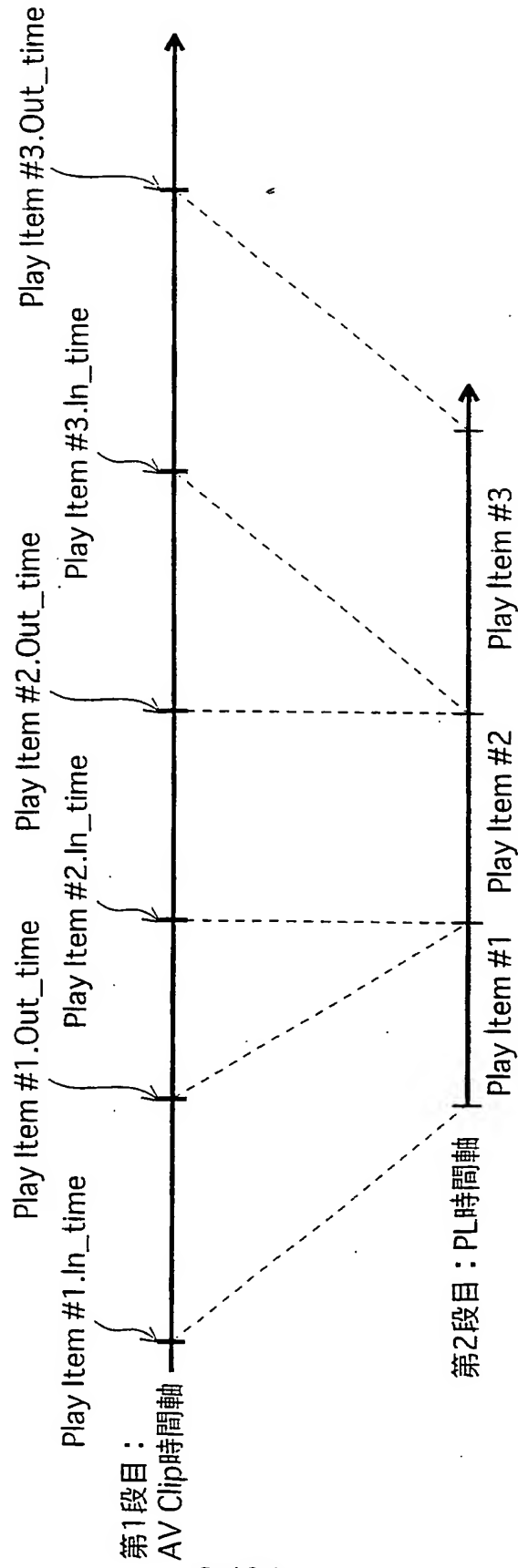


図4

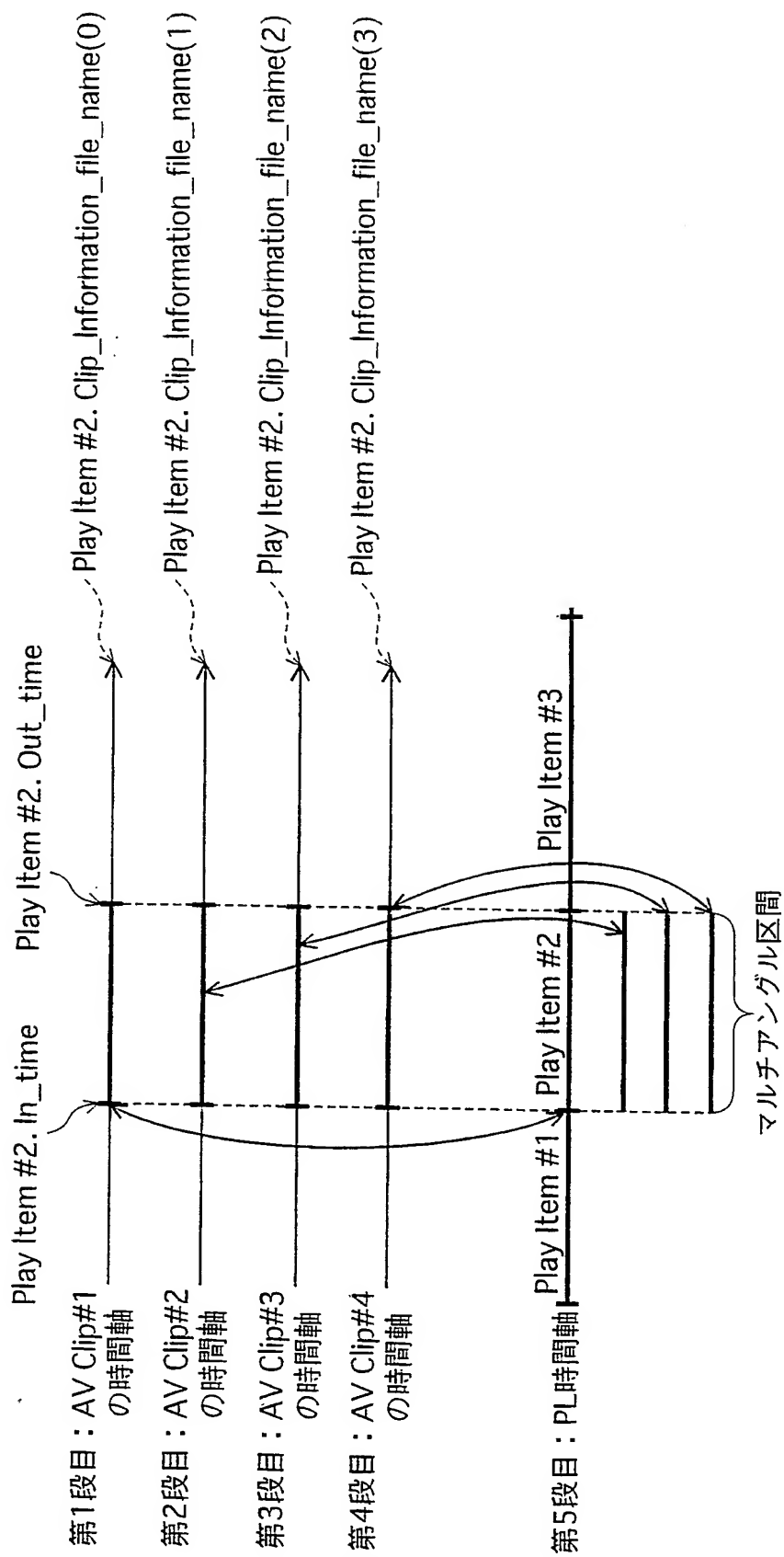


図5

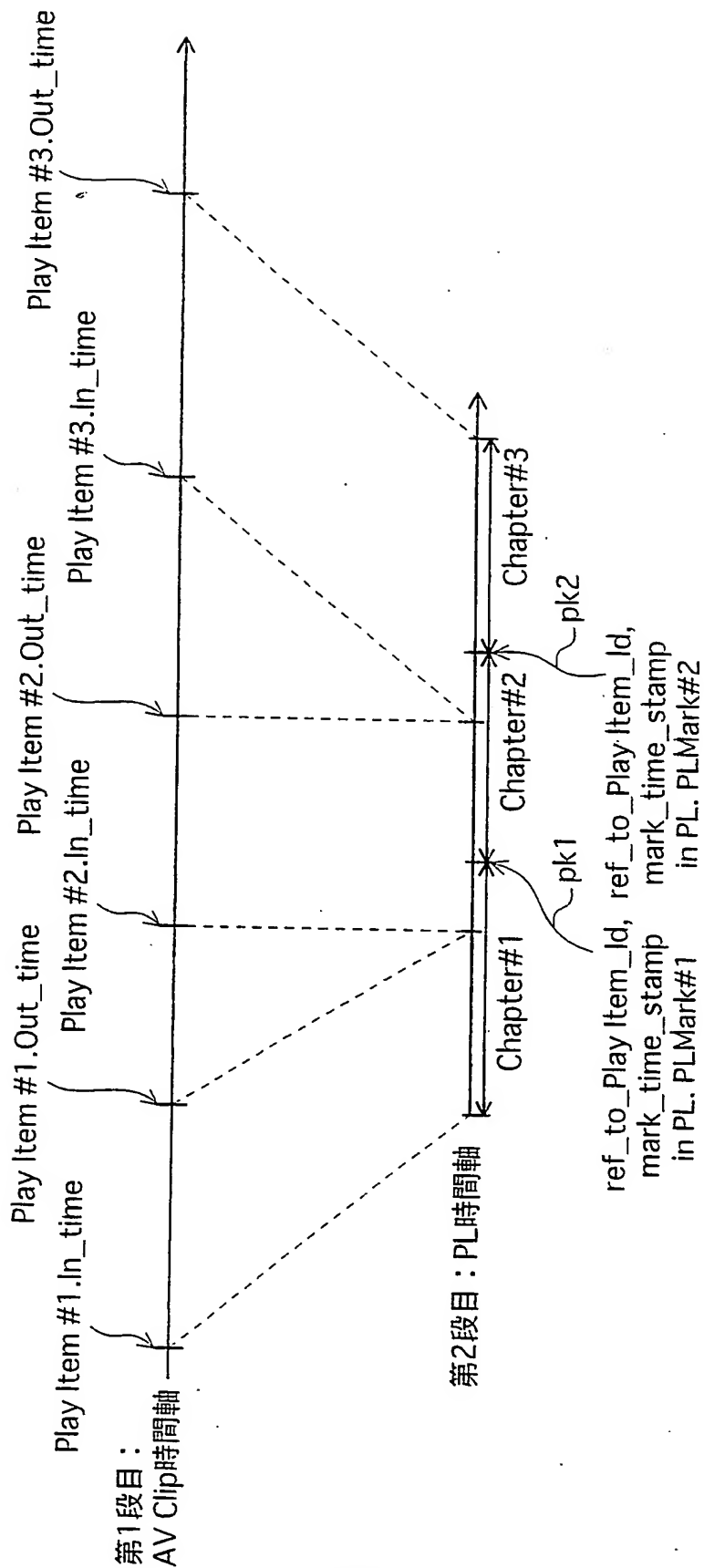


図6

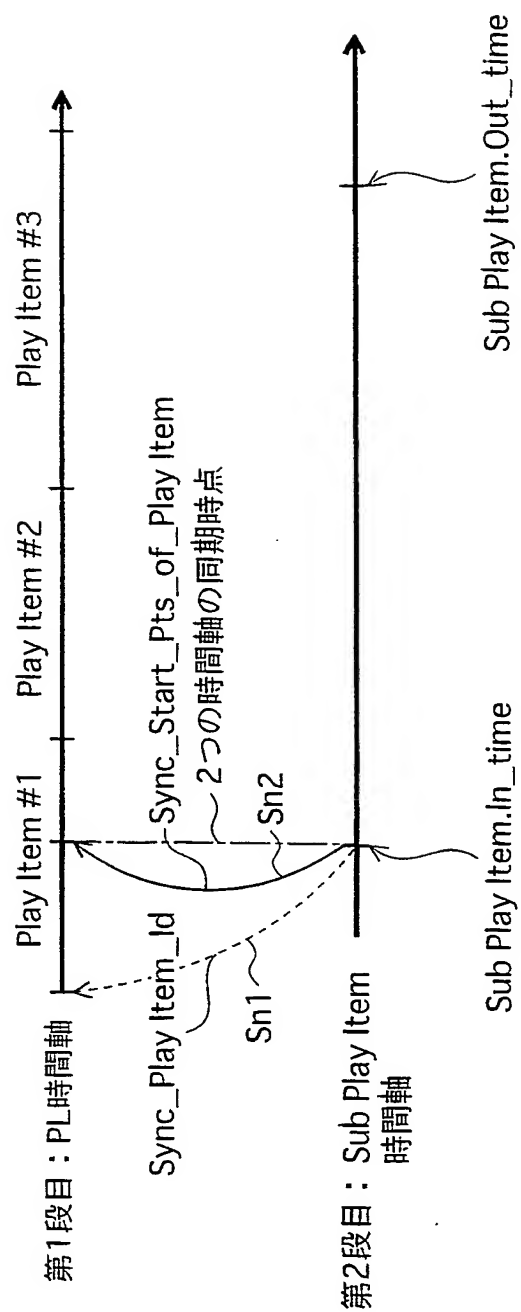
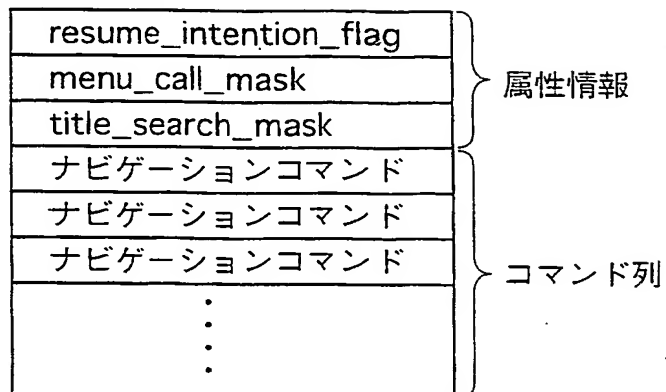


図7

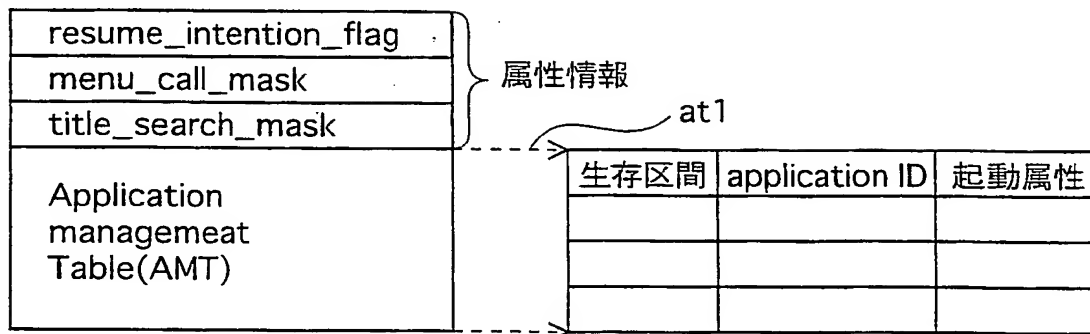
(a)

ZZZZZ.BDMV



(b)

ZZZZZ.BD-J



(c)

Javaアプリケーション

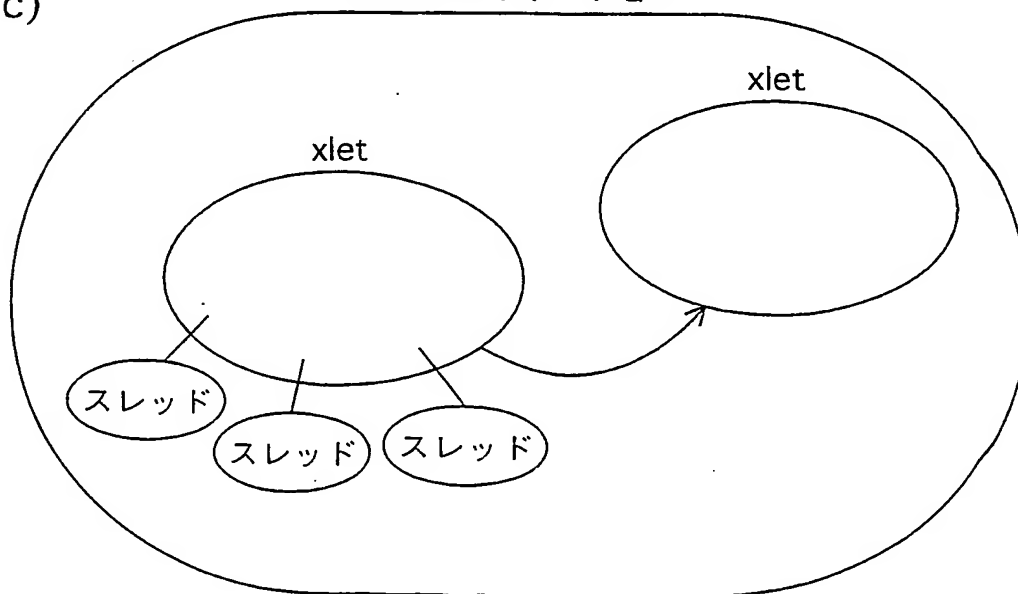


図8

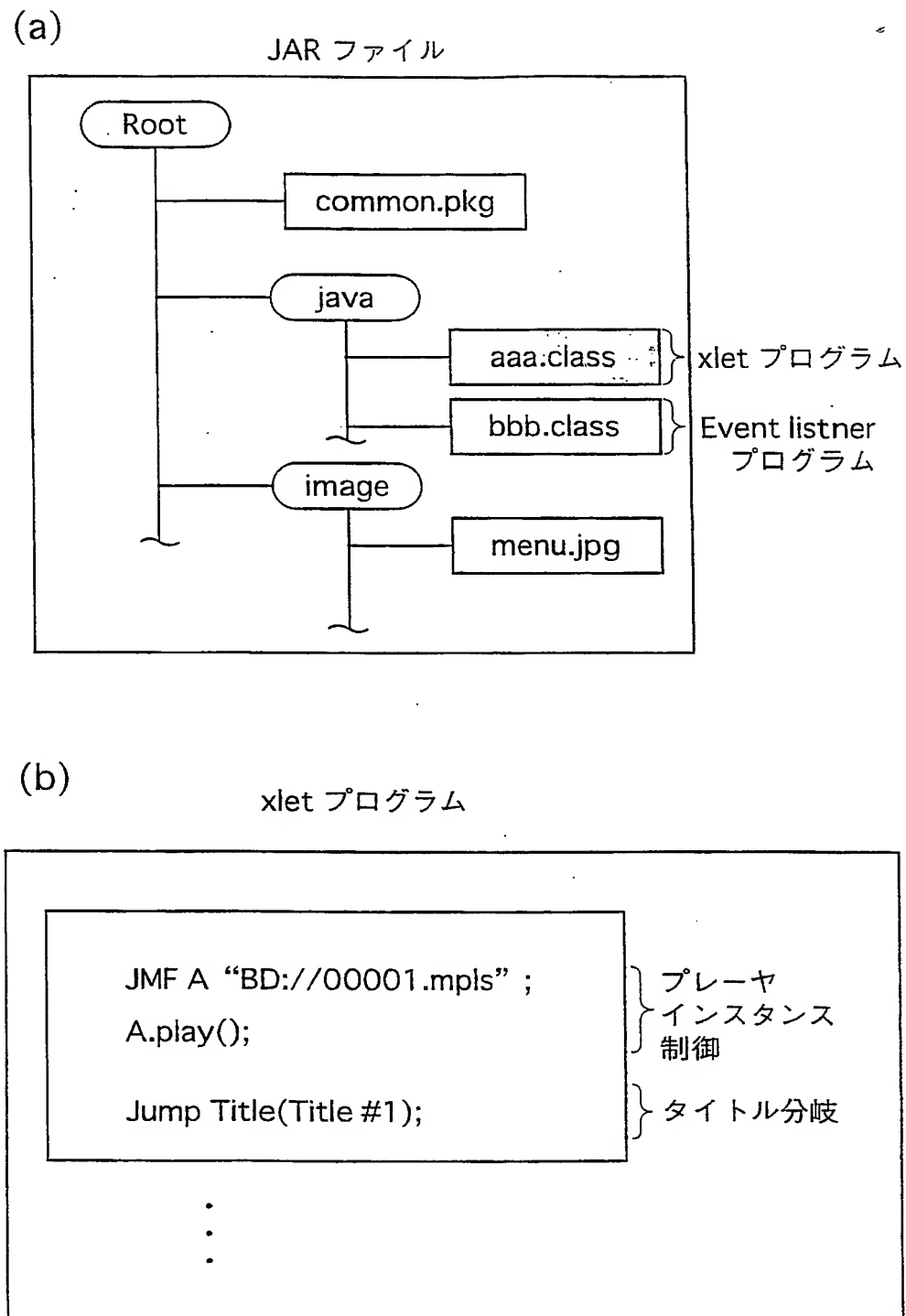


図9

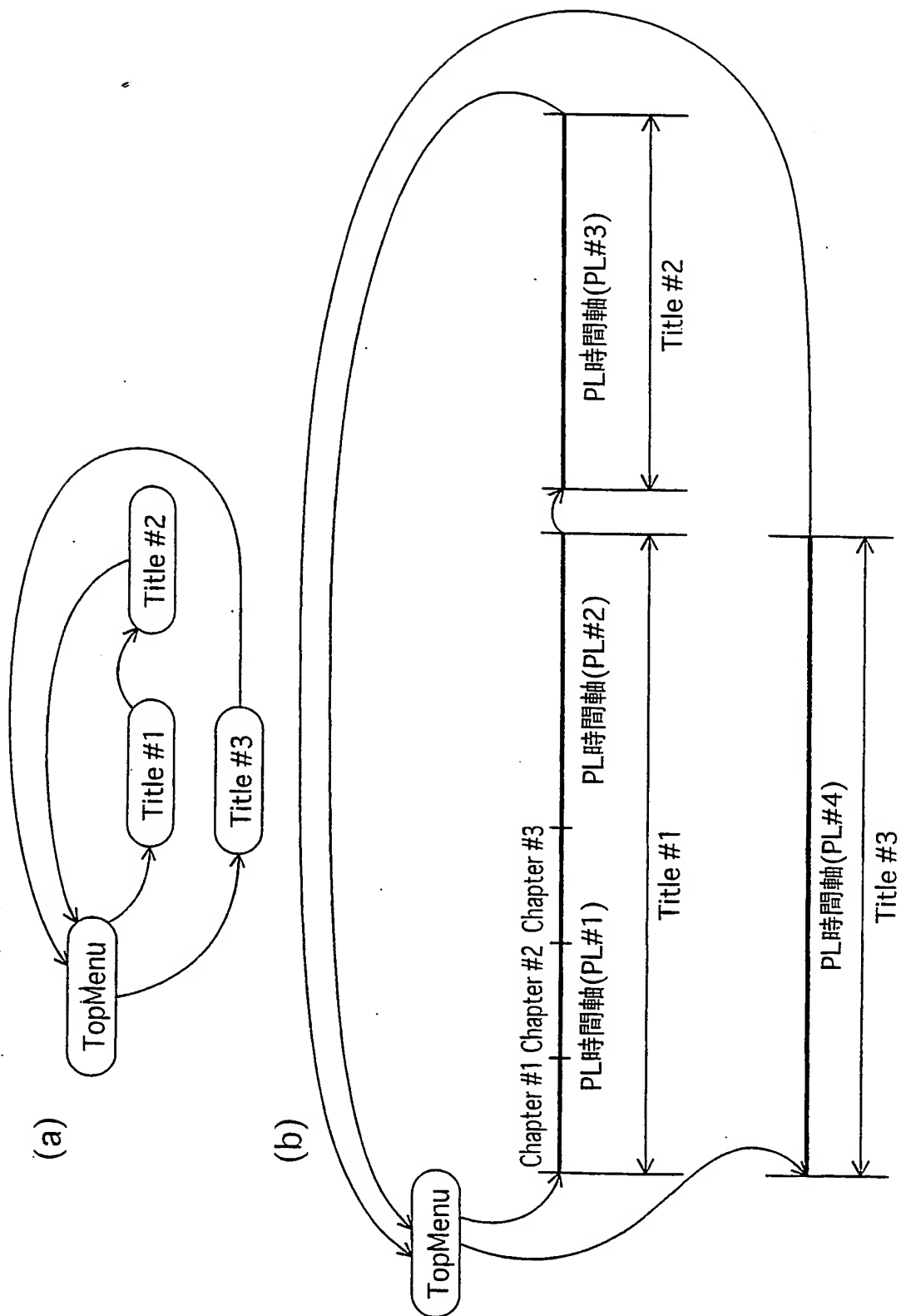


図10

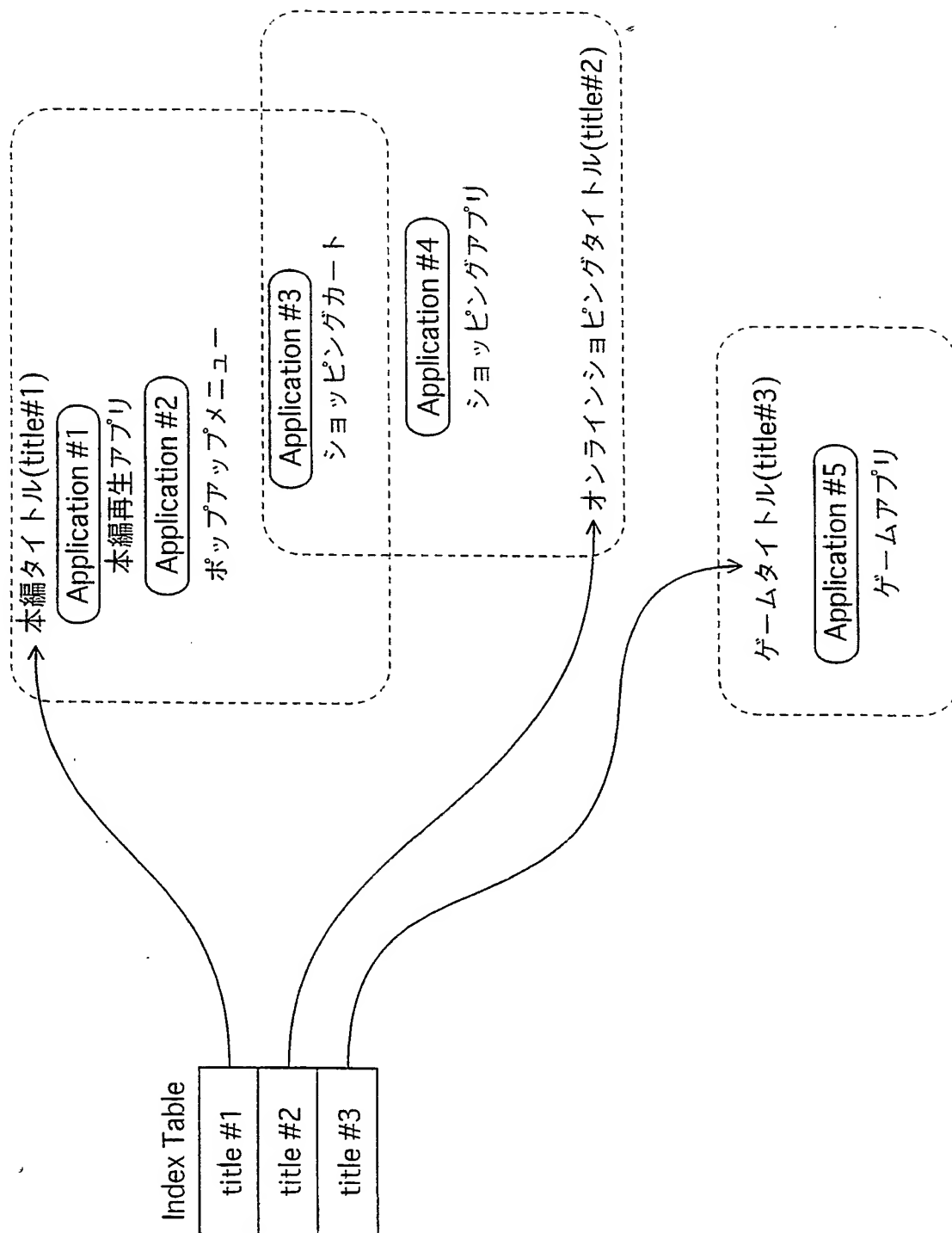


図11

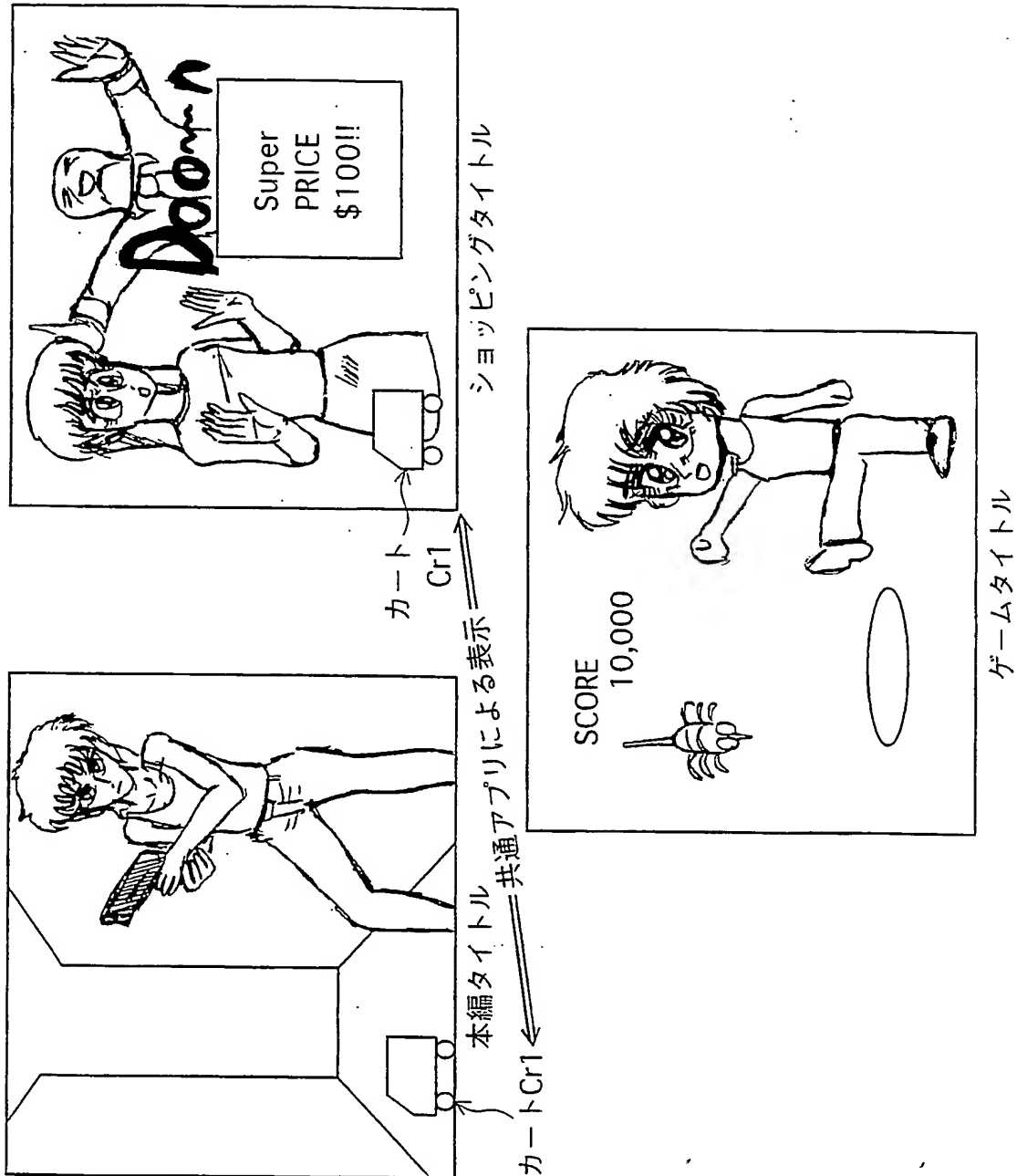
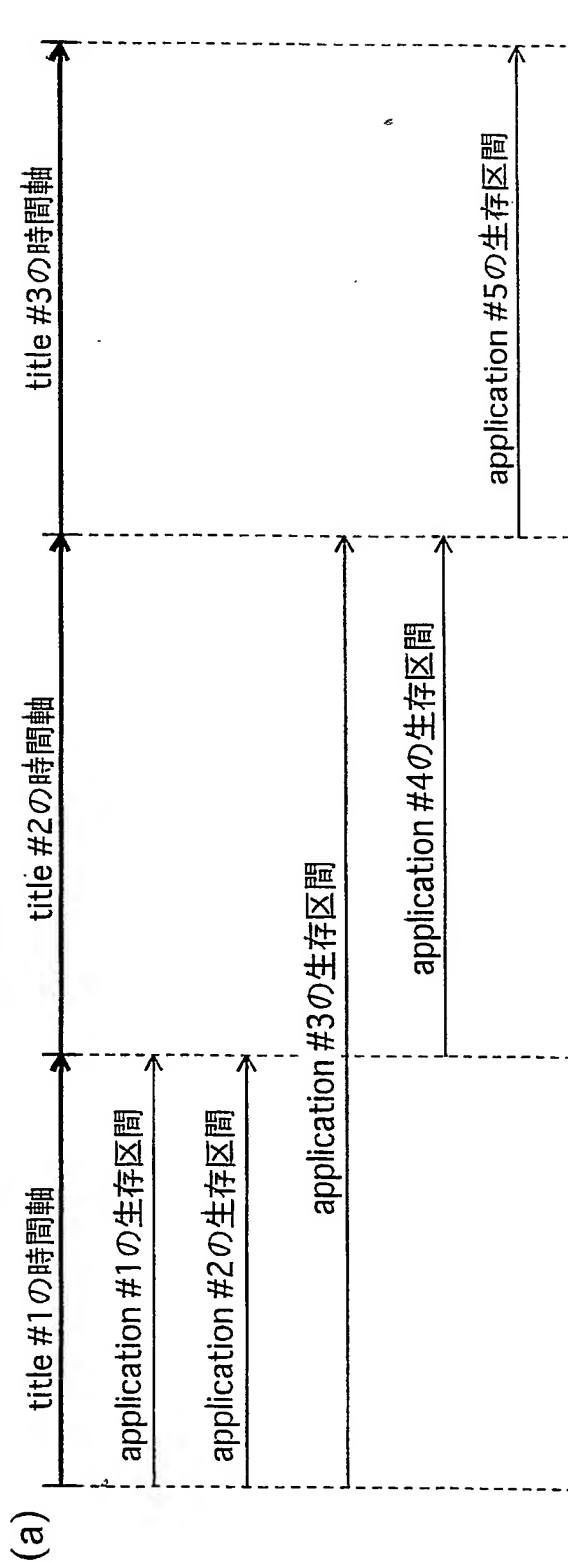


図12



(b)

title #1のアプリケーション管理テーブル			
生存区間	アプリケーションID	起動属性	
title #1	application #1		
title #1	application #2		
title #1	application #3		

title #2のアプリケーション管理テーブル			
生存区間	アプリケーションID	起動属性	
title #2	application #3		
title #2	application #4		

title #3のアプリケーション管理テーブル			
生存区間	アプリケーションID	起動属性	
title #3	application #5		

図13

(a)

title #1のアプリケーション管理テーブル			
生存区間	アプリケーションID	起動属性	
title #1	application #1	AutoRun	
title #1	application #2	Persistent	
title #1	application #3	AutoRun	
title #2のアプリケーション管理テーブル			
生存区間	アプリケーションID	起動属性	
title #2	application #3	Persistent	

(b)

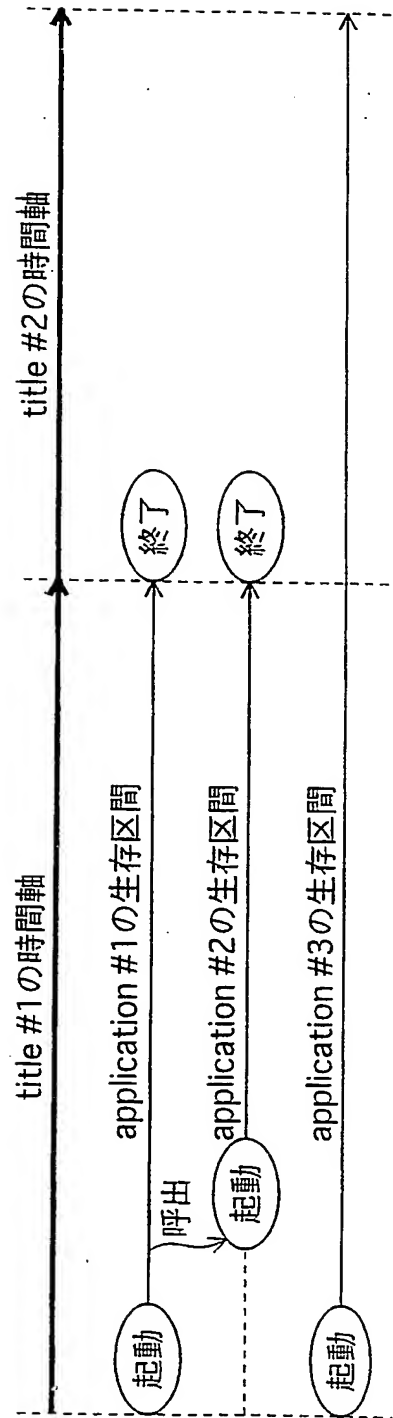


図14

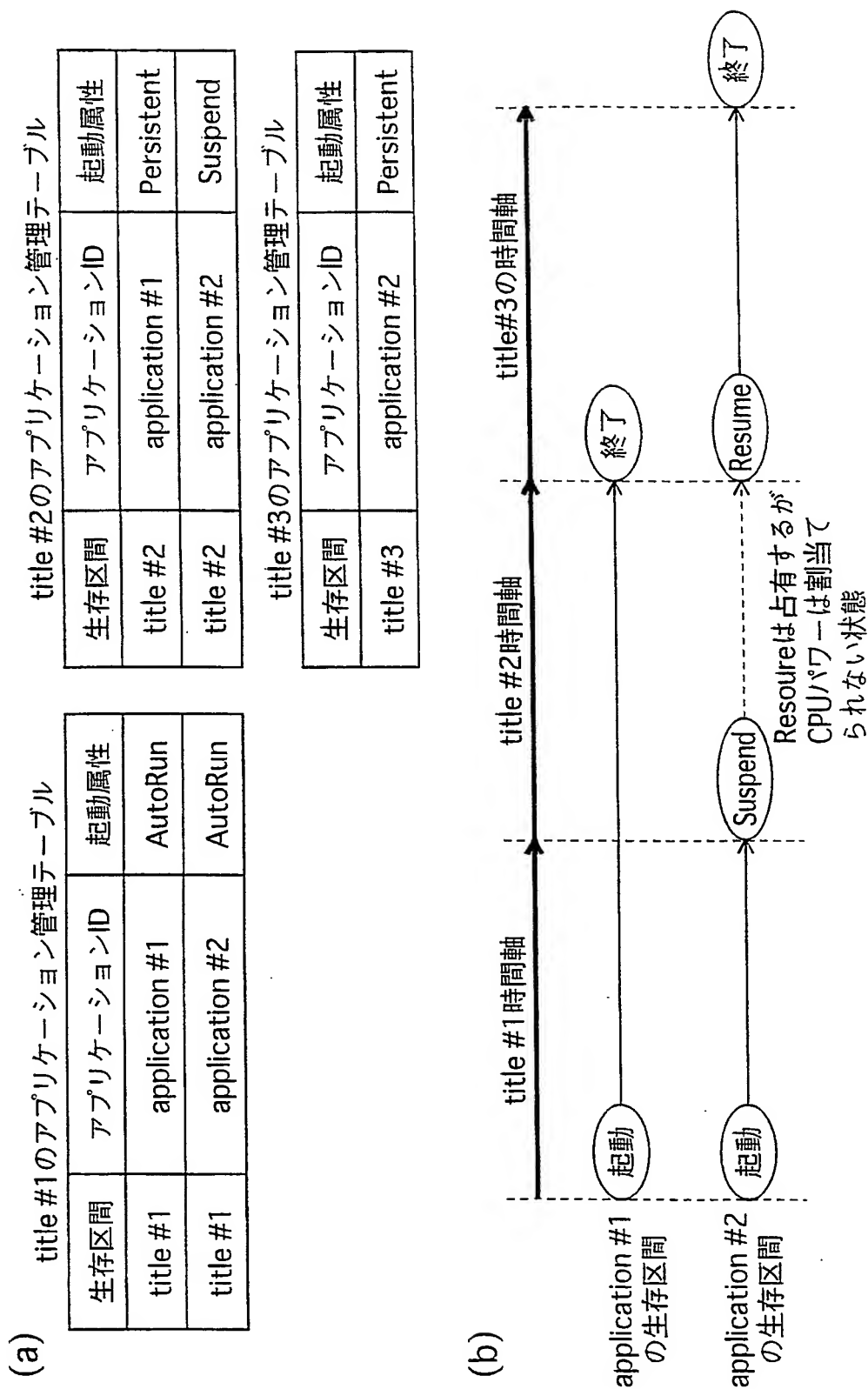


図15

起動属性によるアプリケーション状態の変化

		起動属性		
		Persistent	AutoRun	Suspend
直前タイトル における アプリケーション の状態	非起動	何もせず、 状態継続	アプリケーション を起動する	何もせず、 状態継続
	起動中	何もせず、 状態継続	何もせず、 状態継続	サスペンドする
	Suspend	レジュームする	レジュームする	何もせず、 状態継続

図16

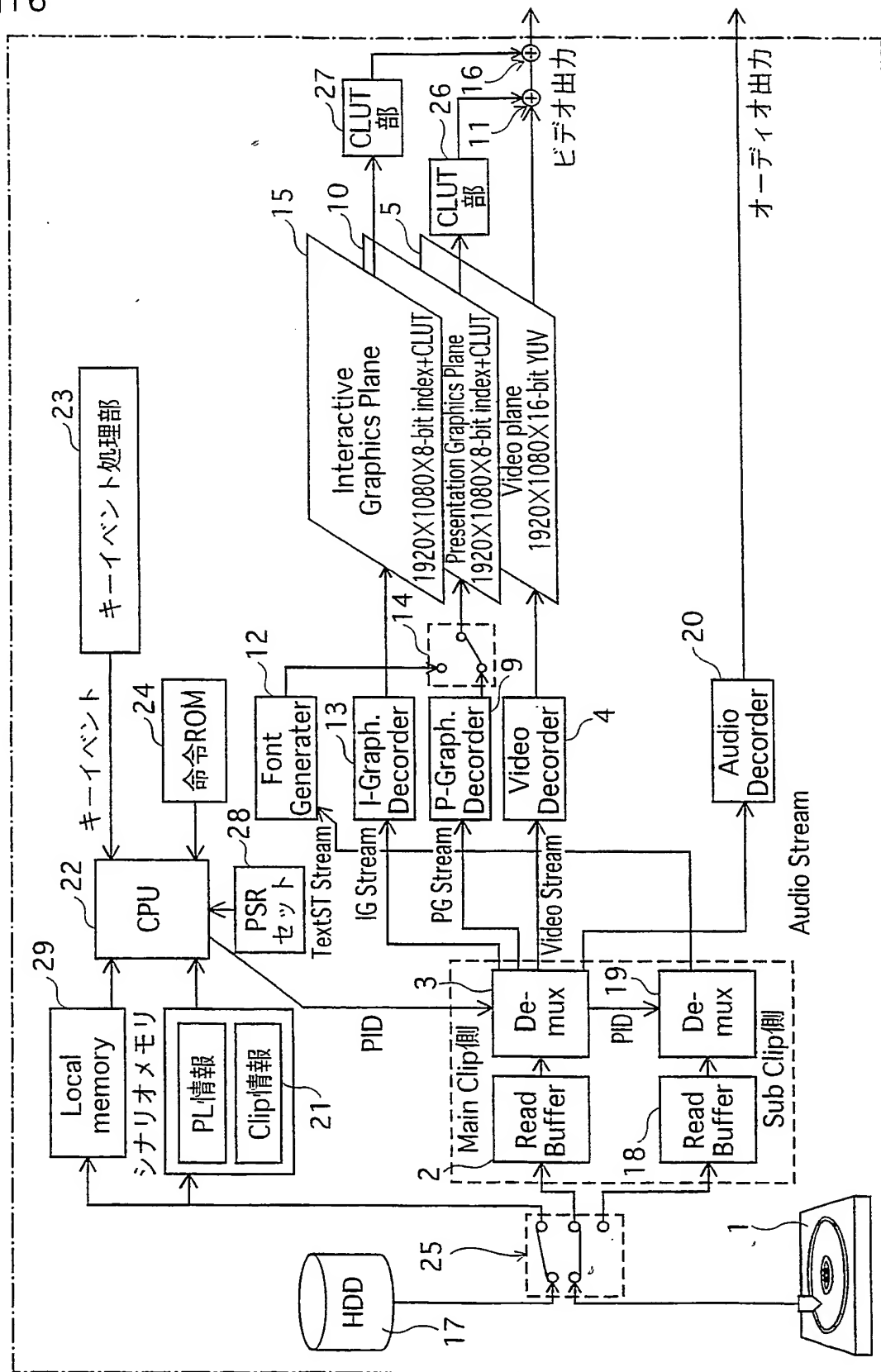


図17

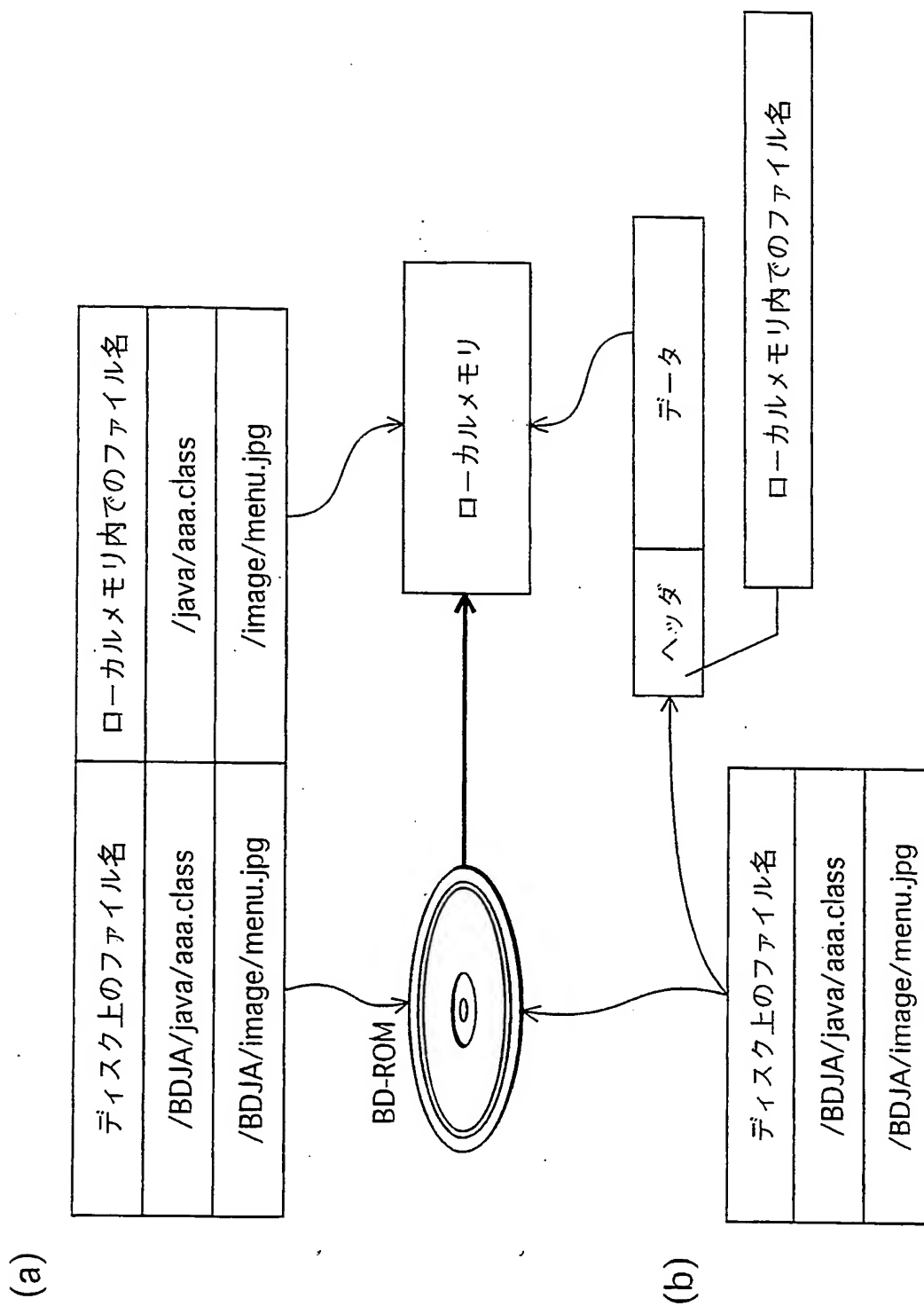


図18

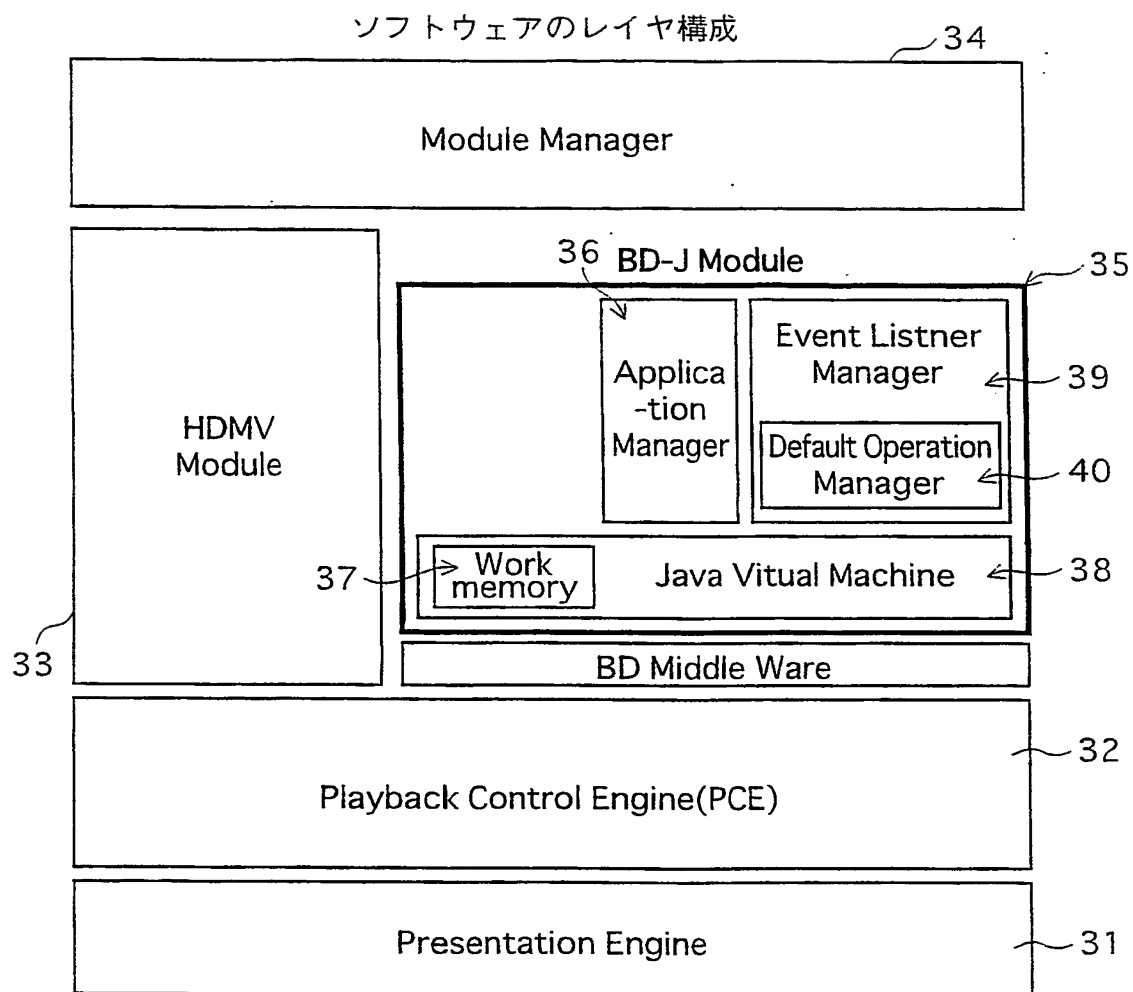


図19

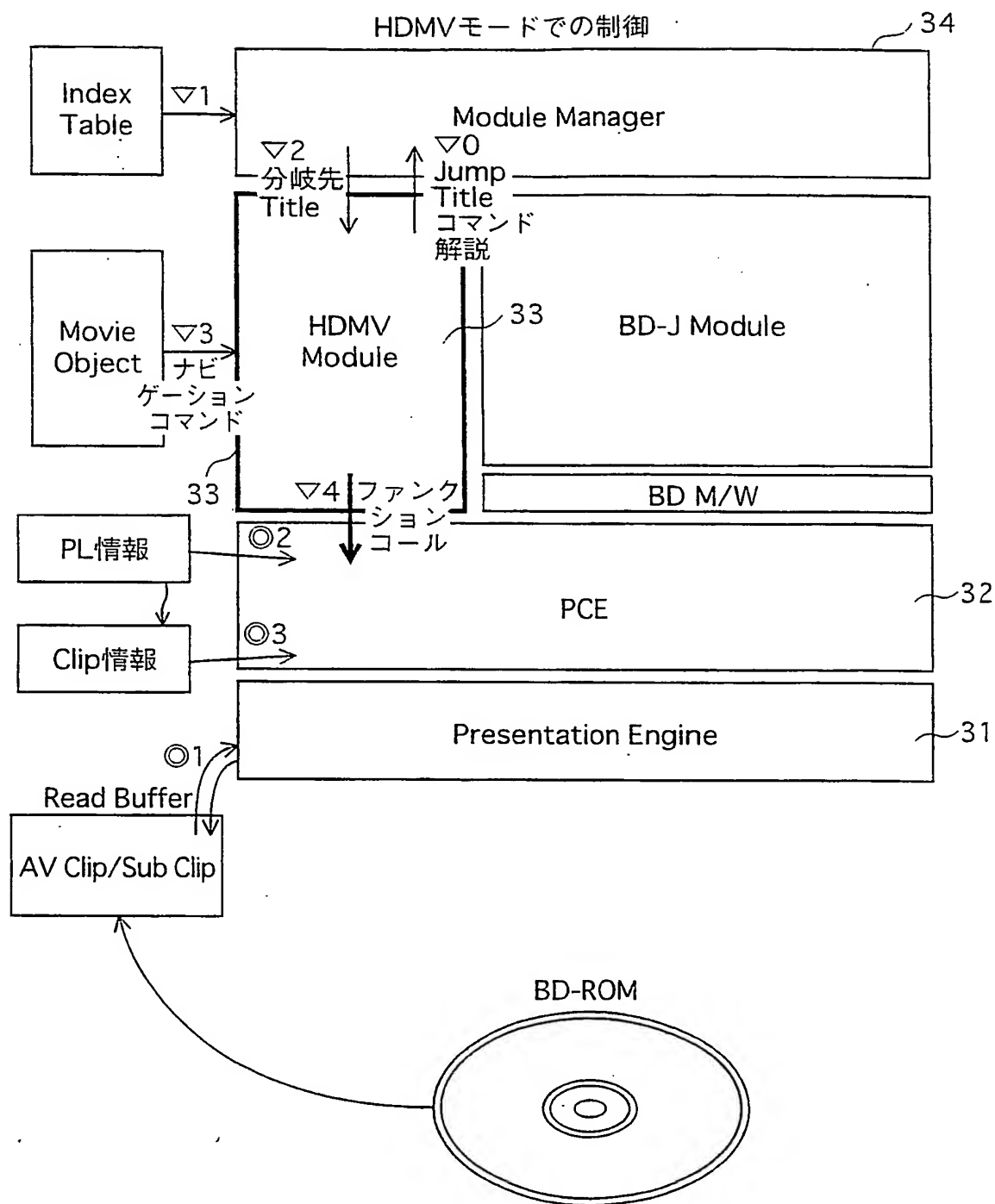


図20

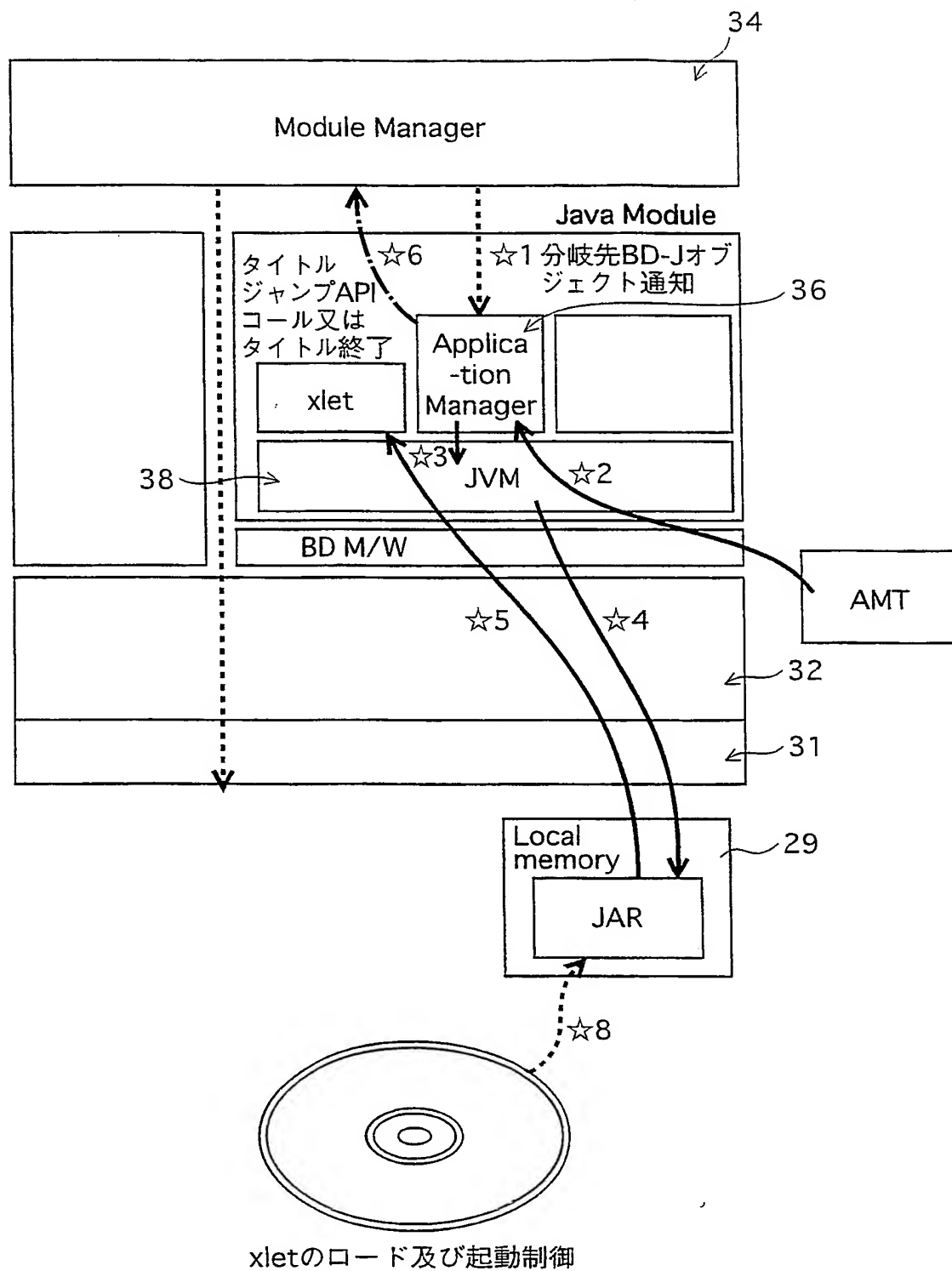
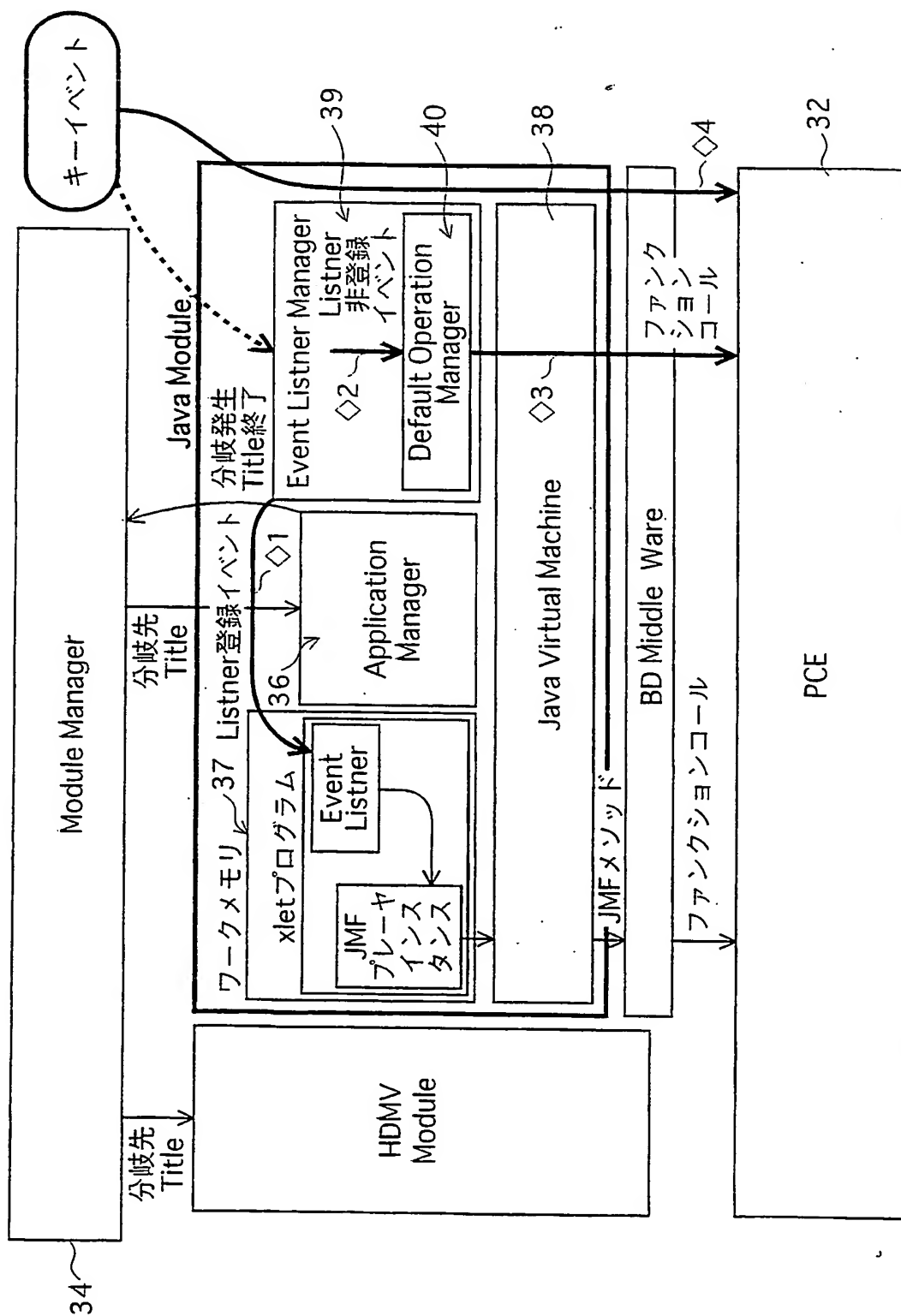


图21



xlet実行時の制御

図22

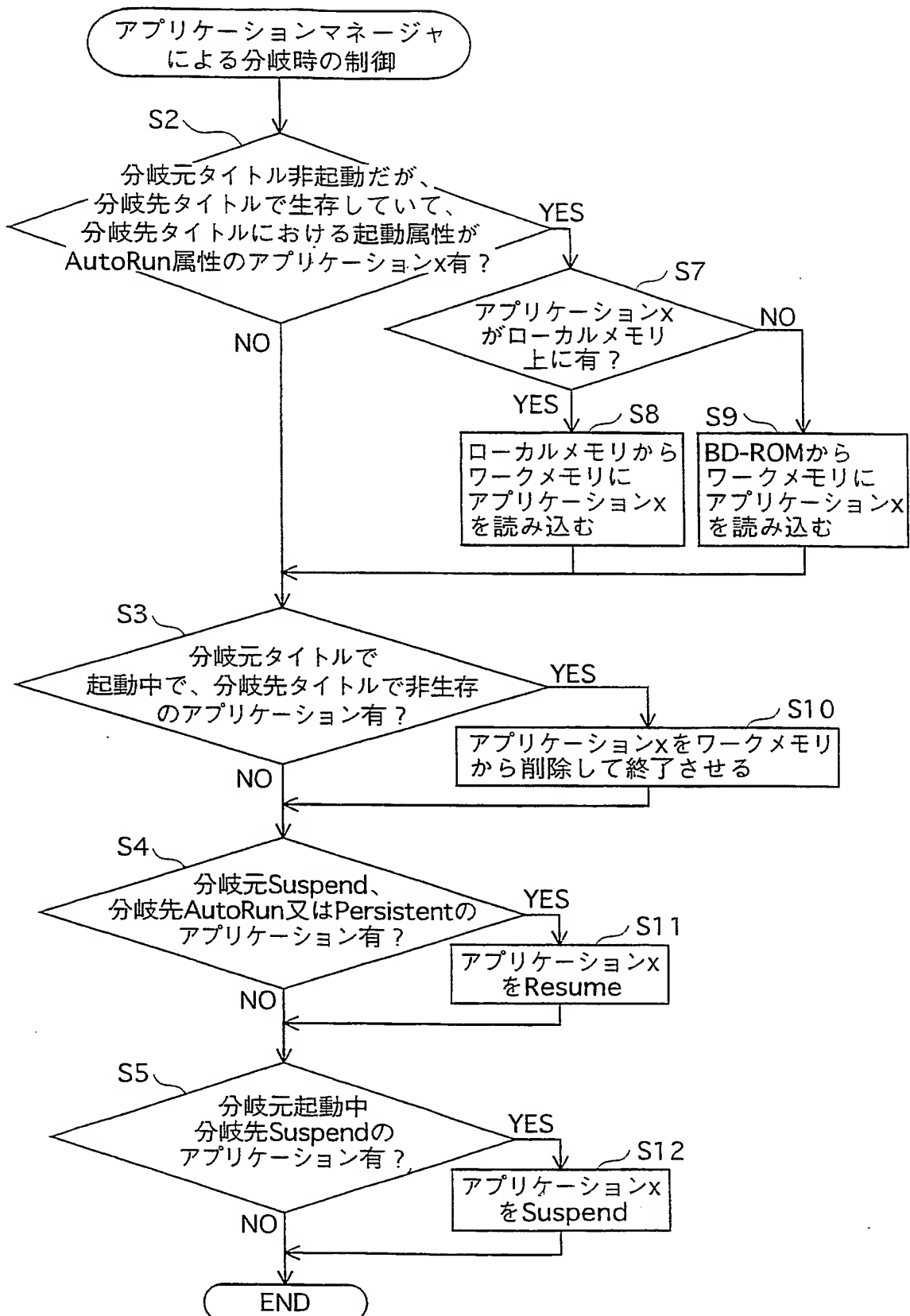


図23

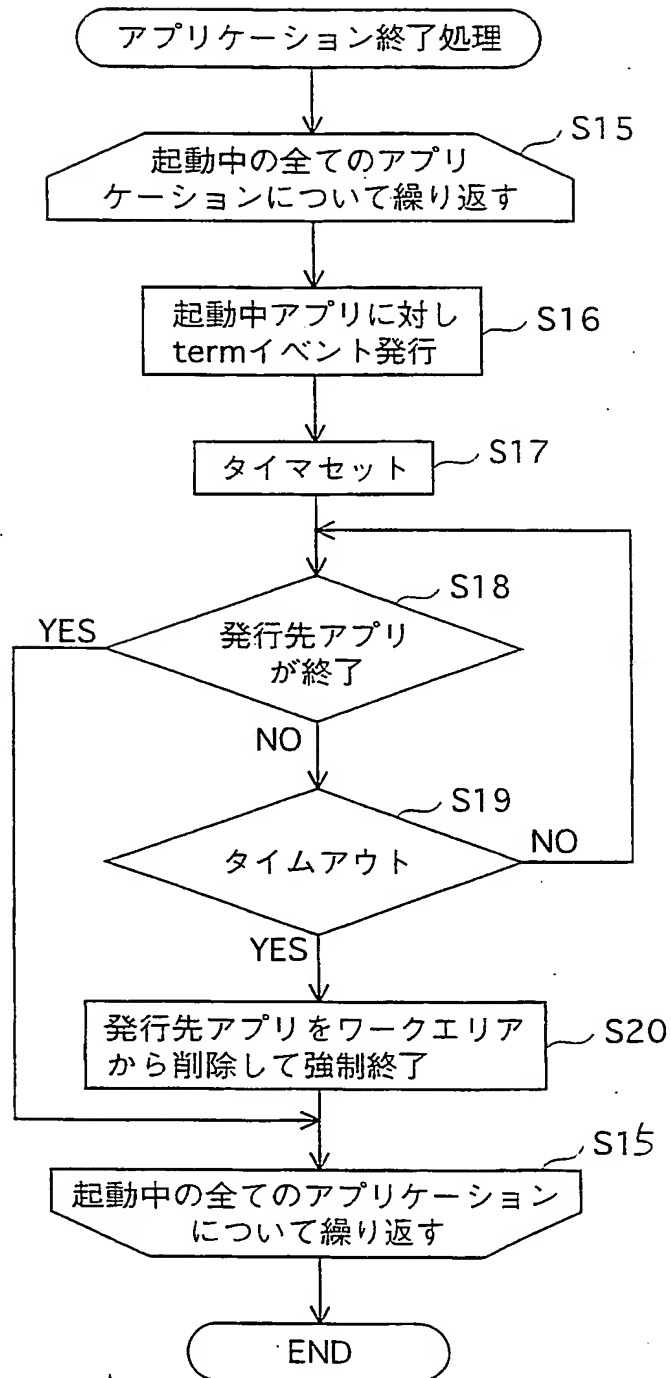


図24

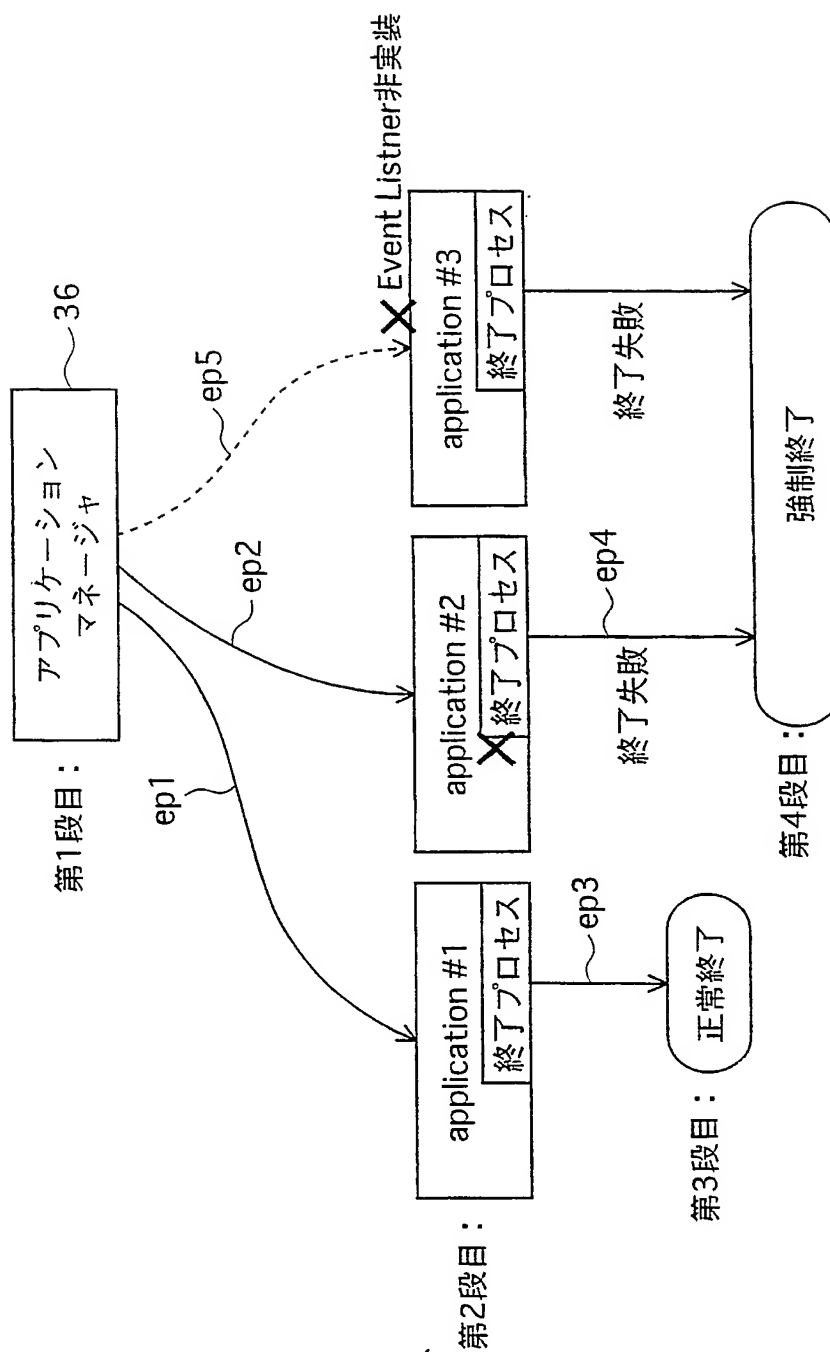
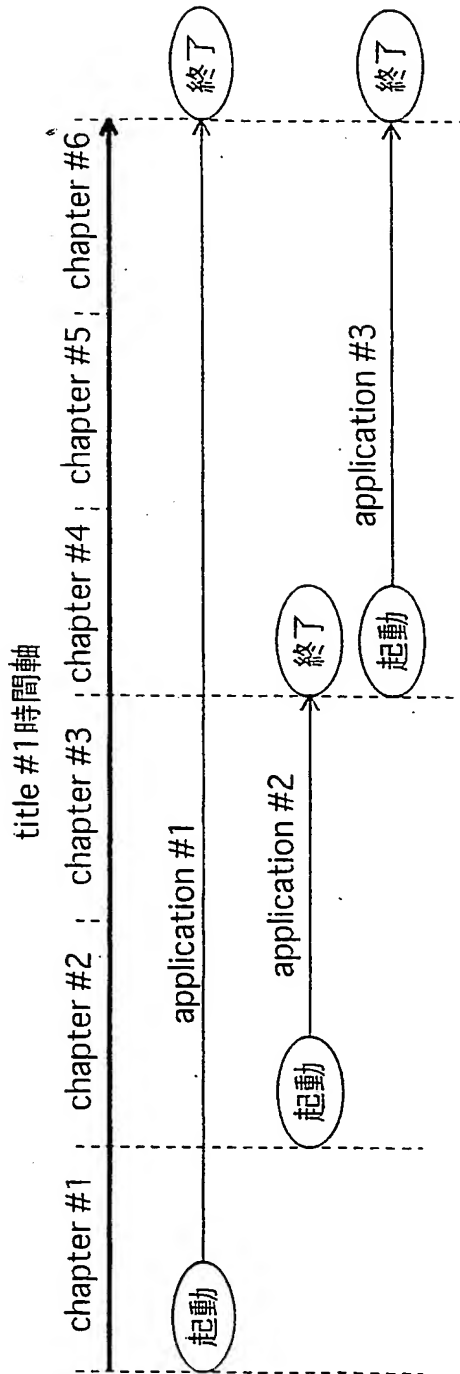


図25

アプリケーション管理テーブル

生存区間	アプリケーションID	起動属性	起動優先度
title #1	application #1	AutoRun	255
title #1 : chapter #2-#3	application #2	AutoRun	128
title #1 : chapter #4-#6	application #3	AutoRun	128

(a)



(b)

図26

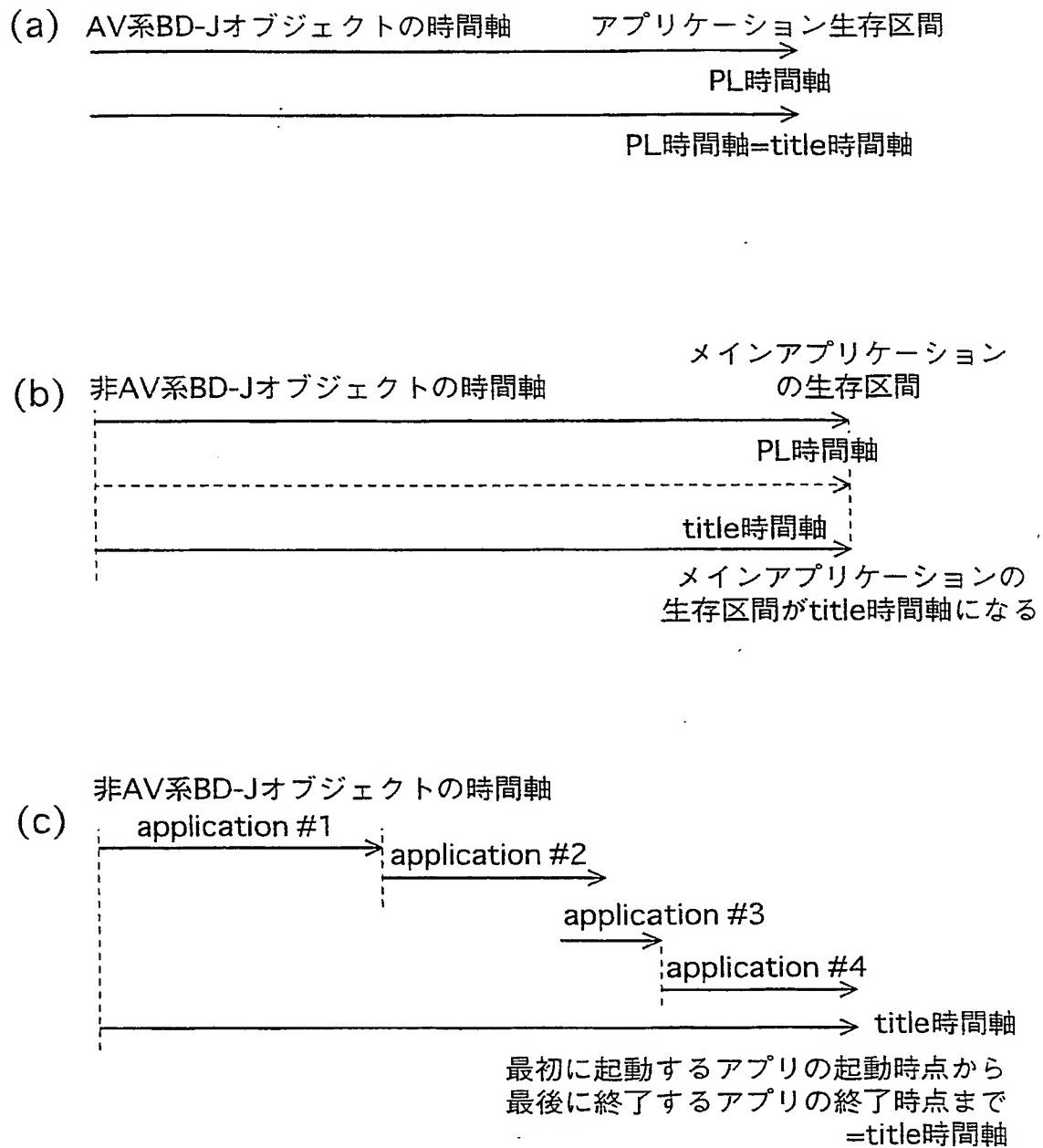


図27

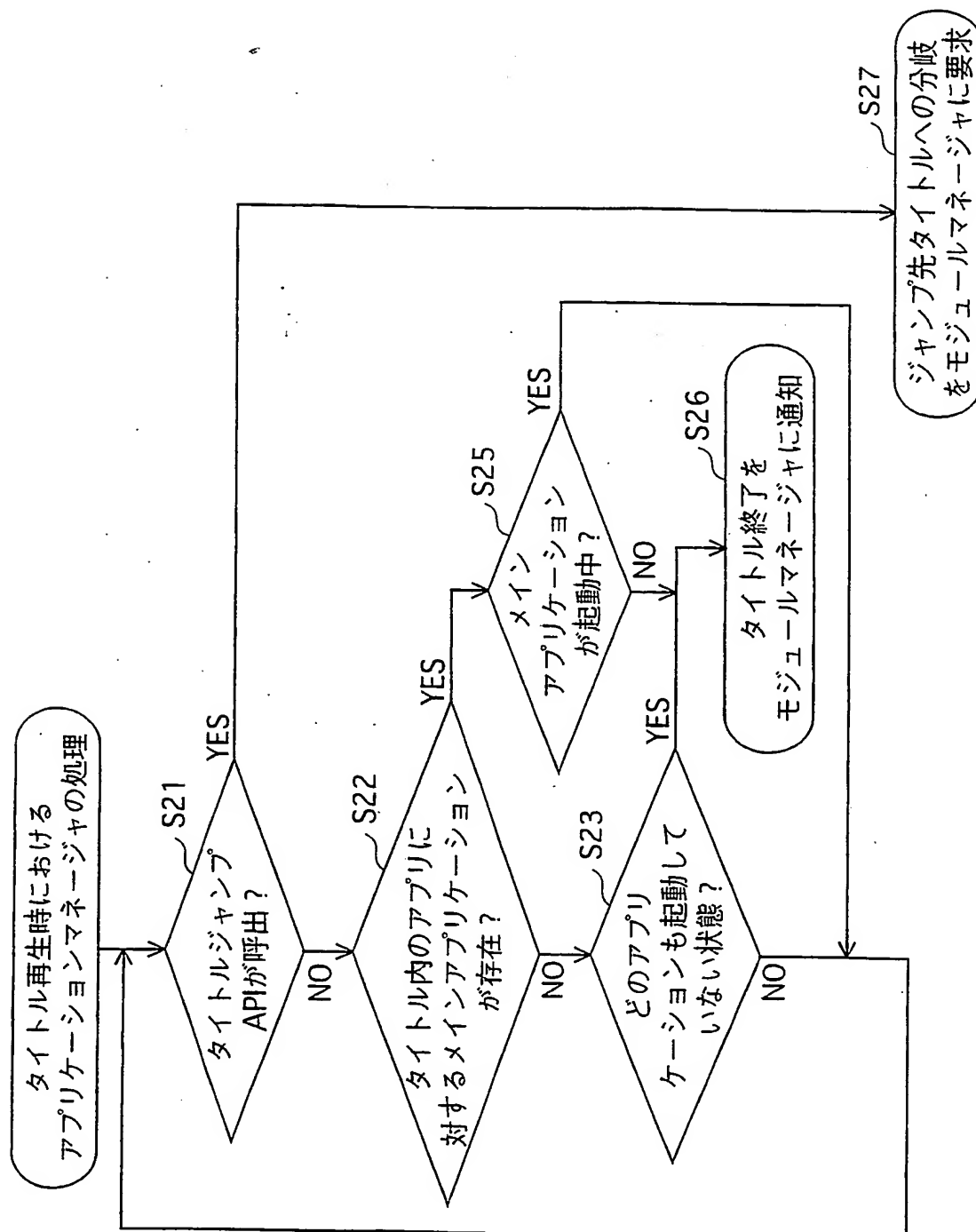
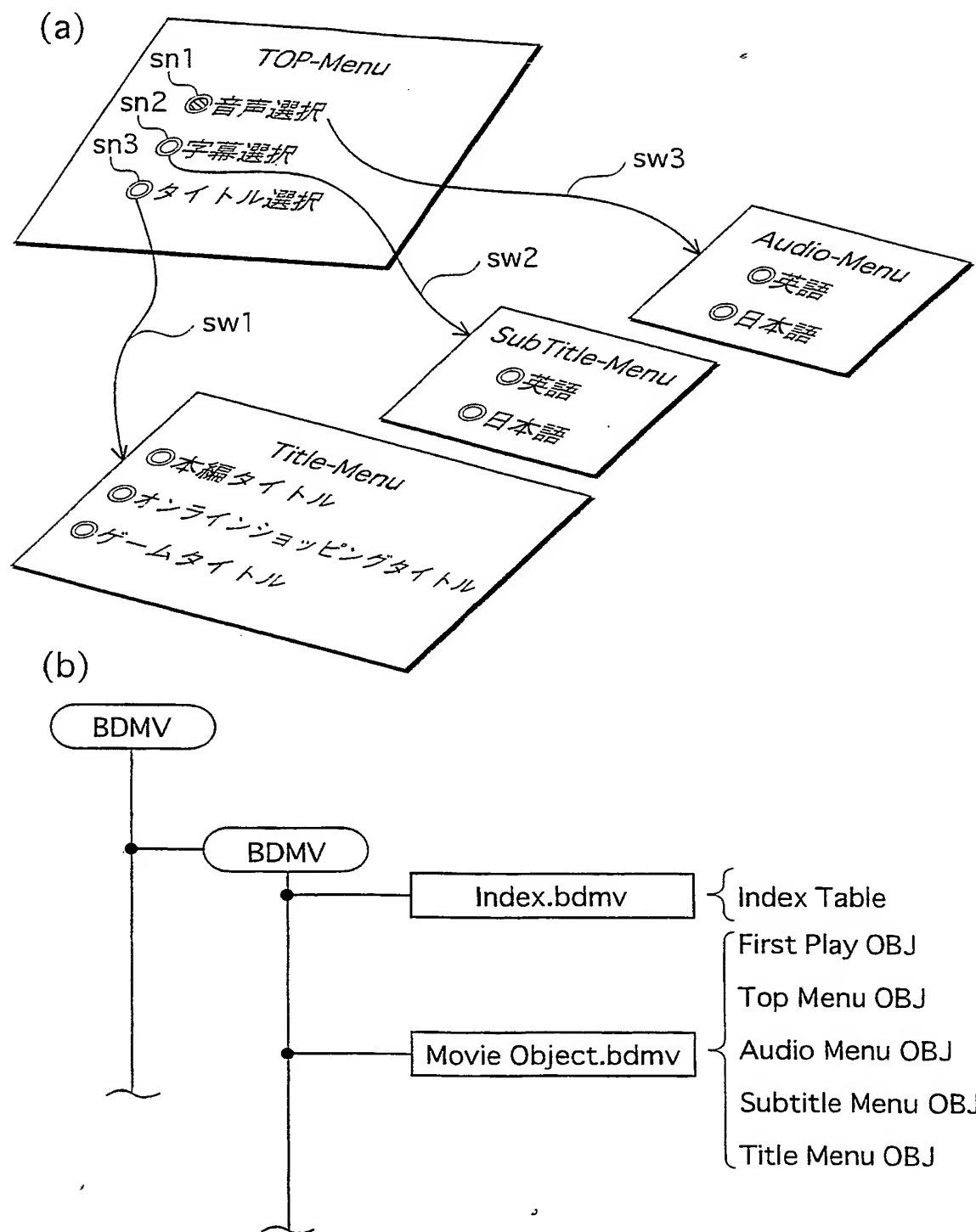


図28



29

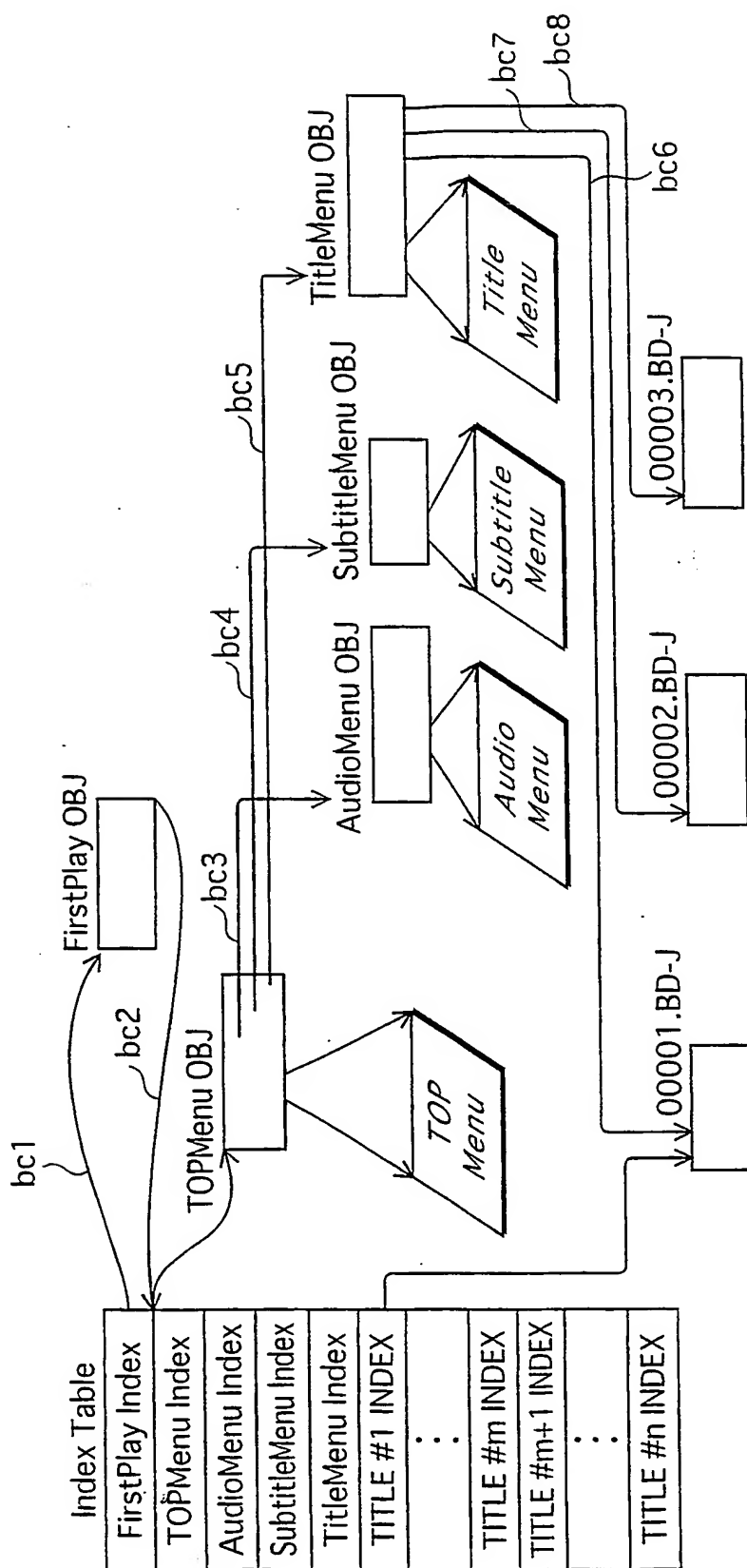
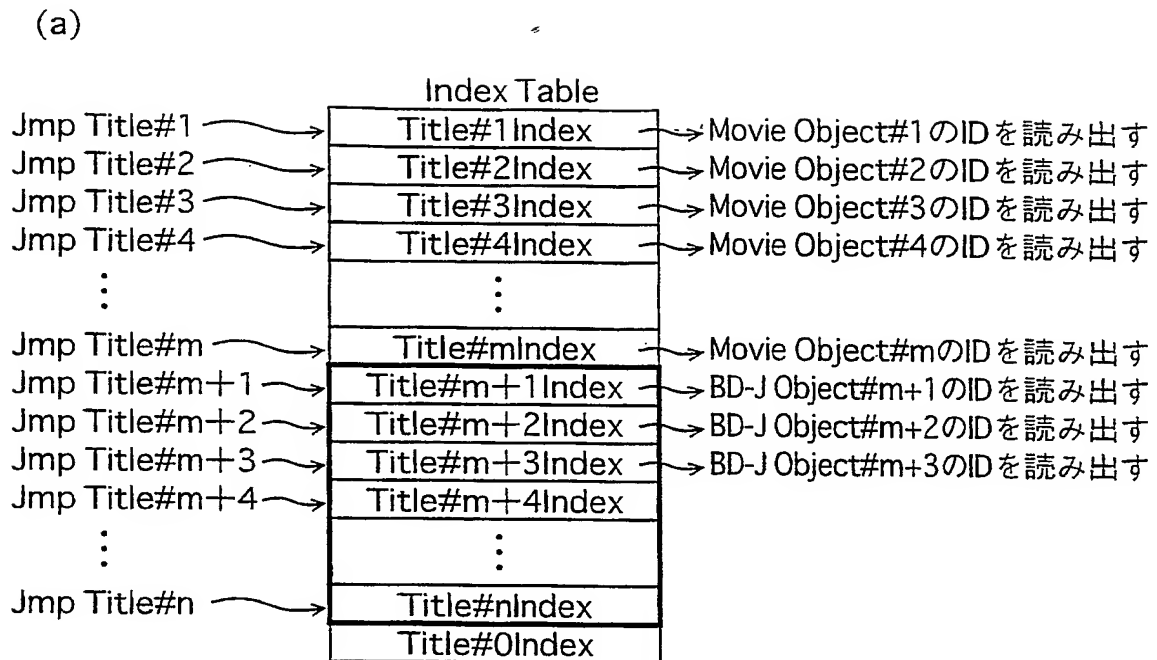


図30



(b) BD-Jオブジェクト実行時の例外処理

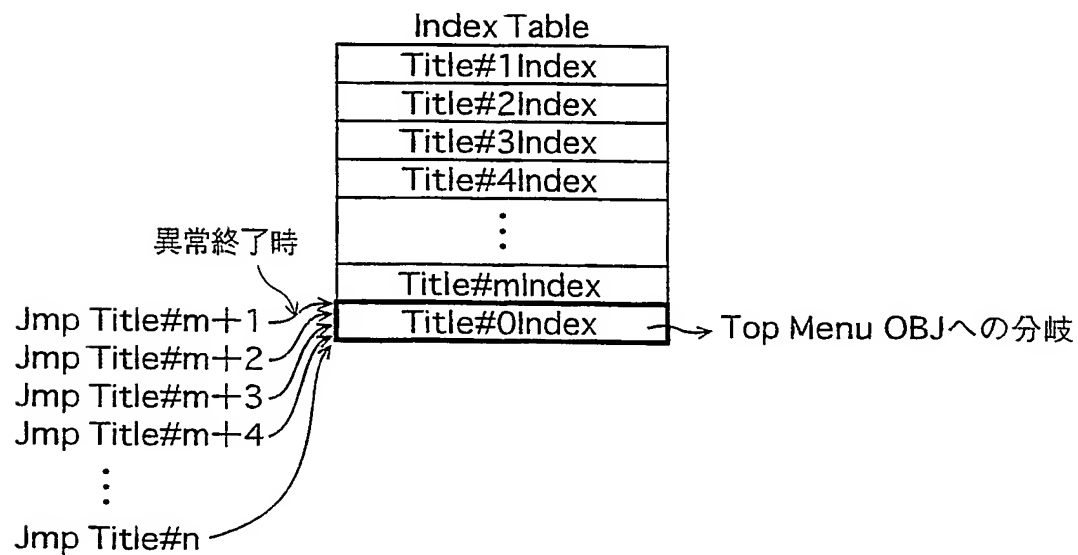


図31

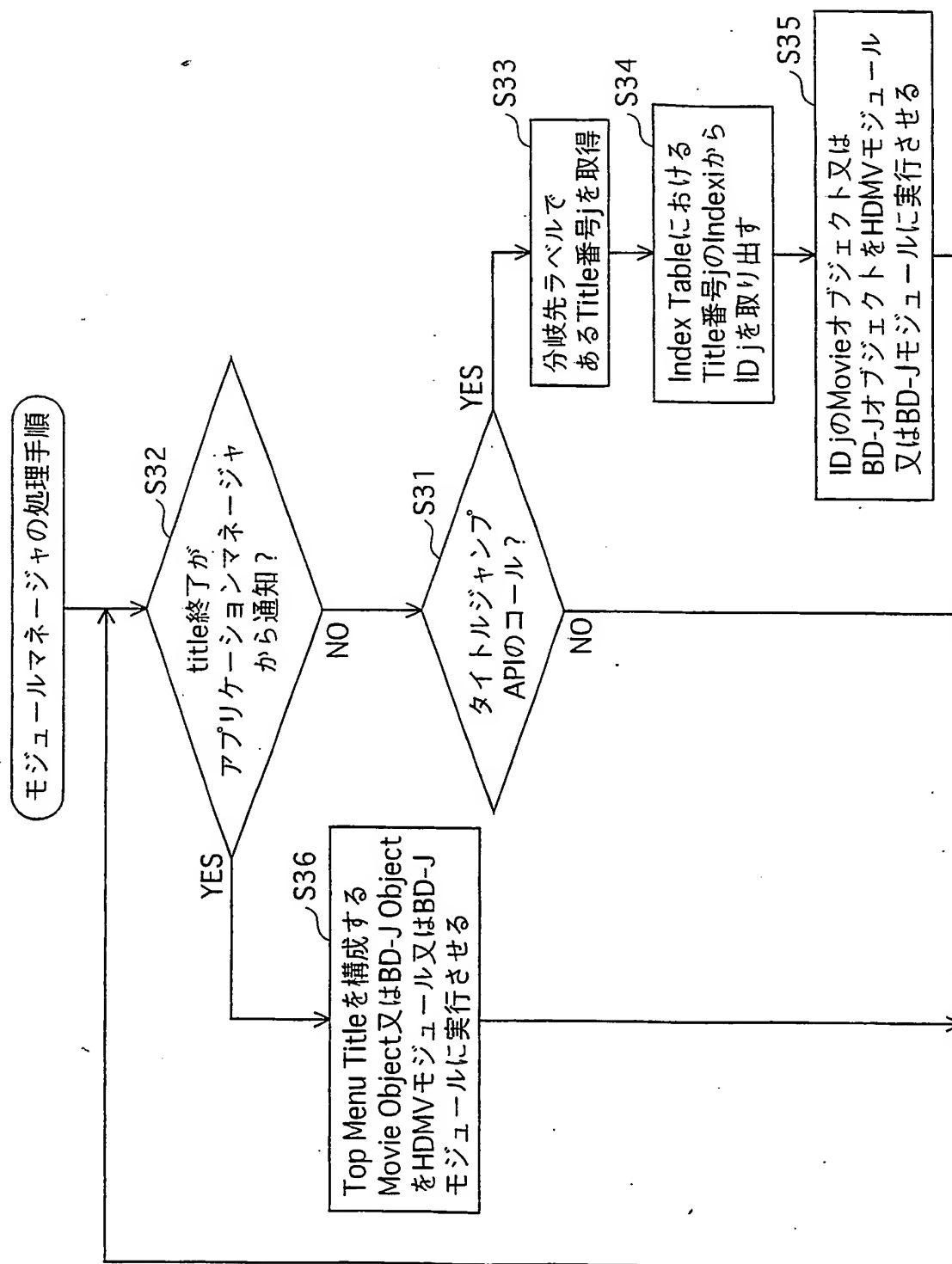


図32

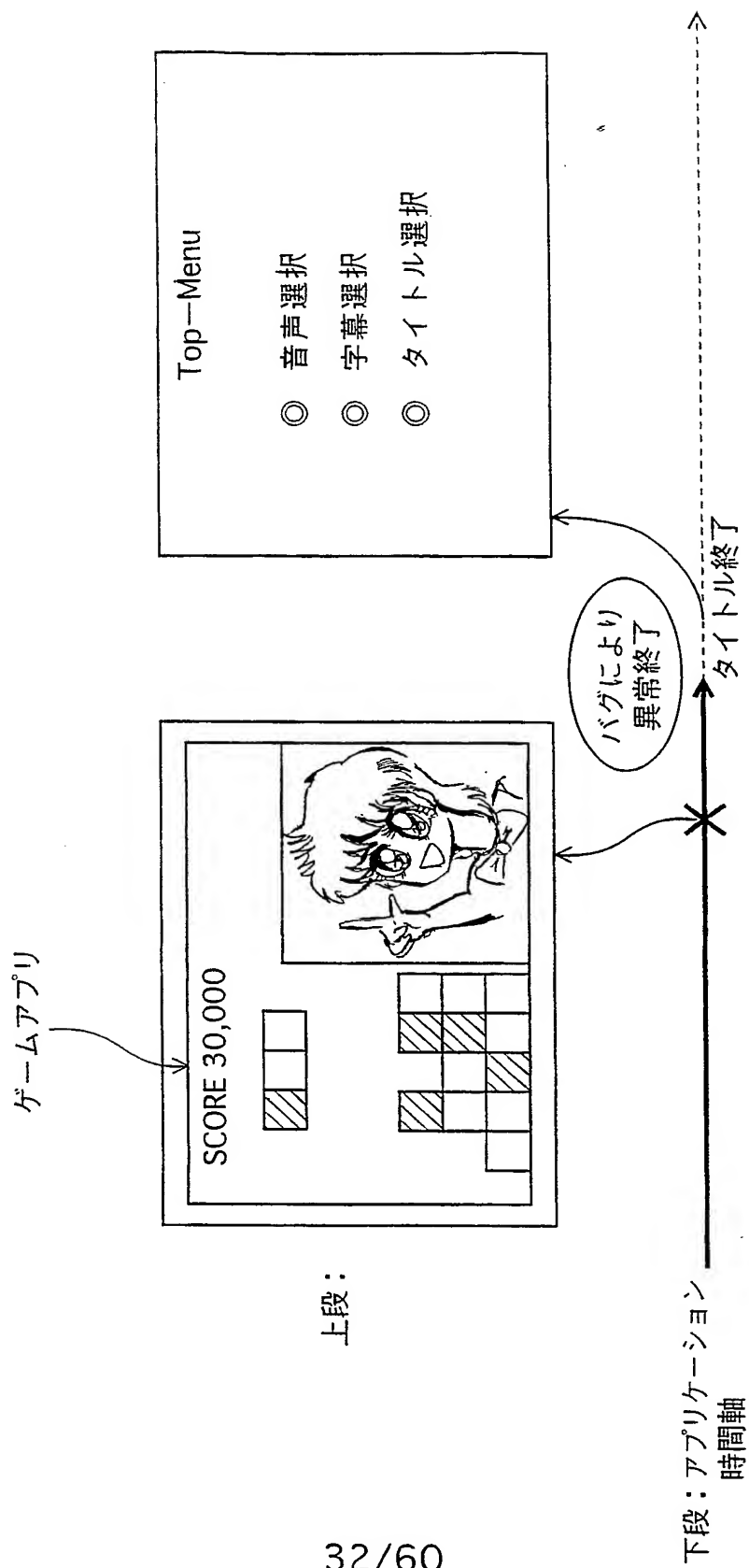


図33

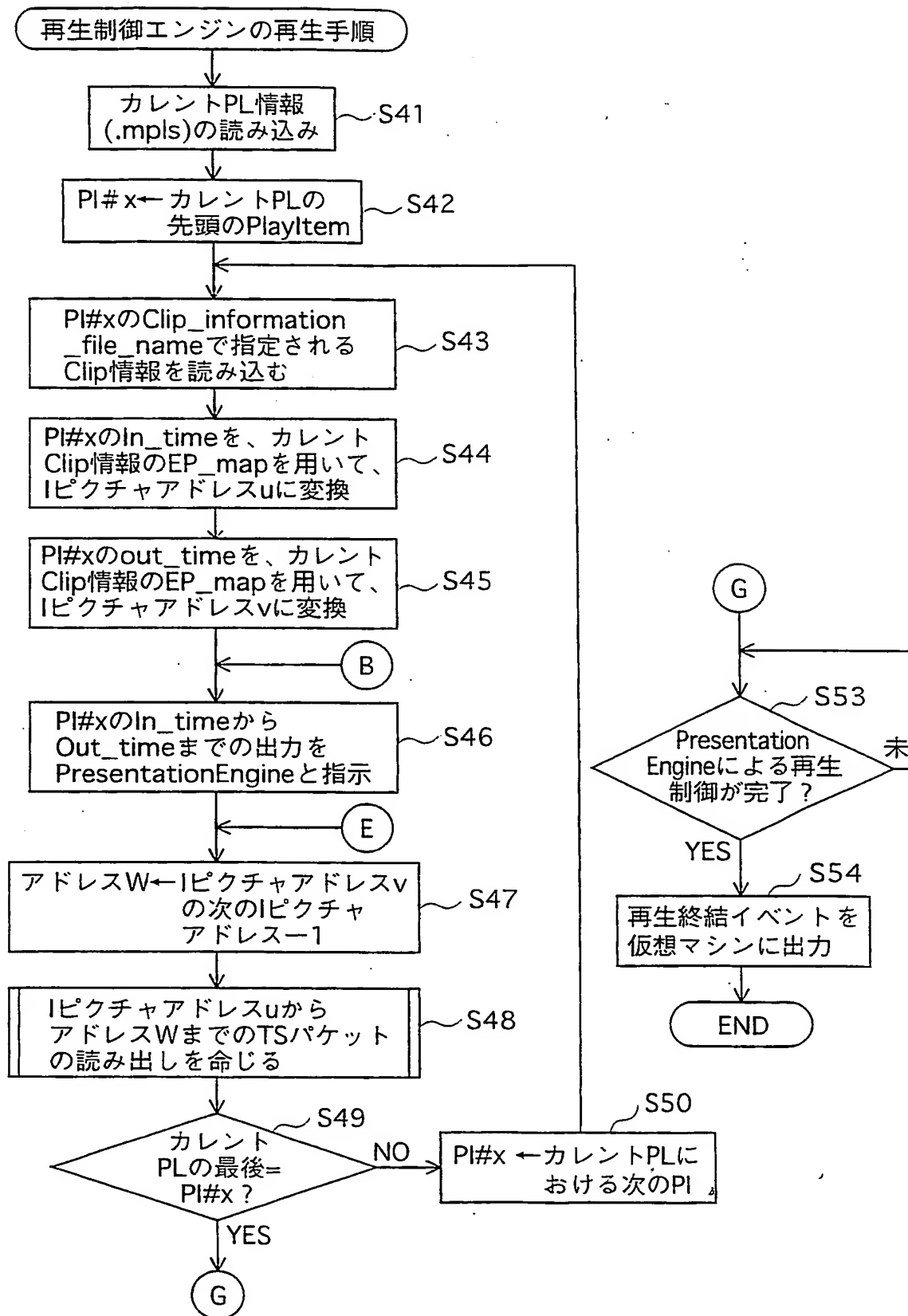


図34

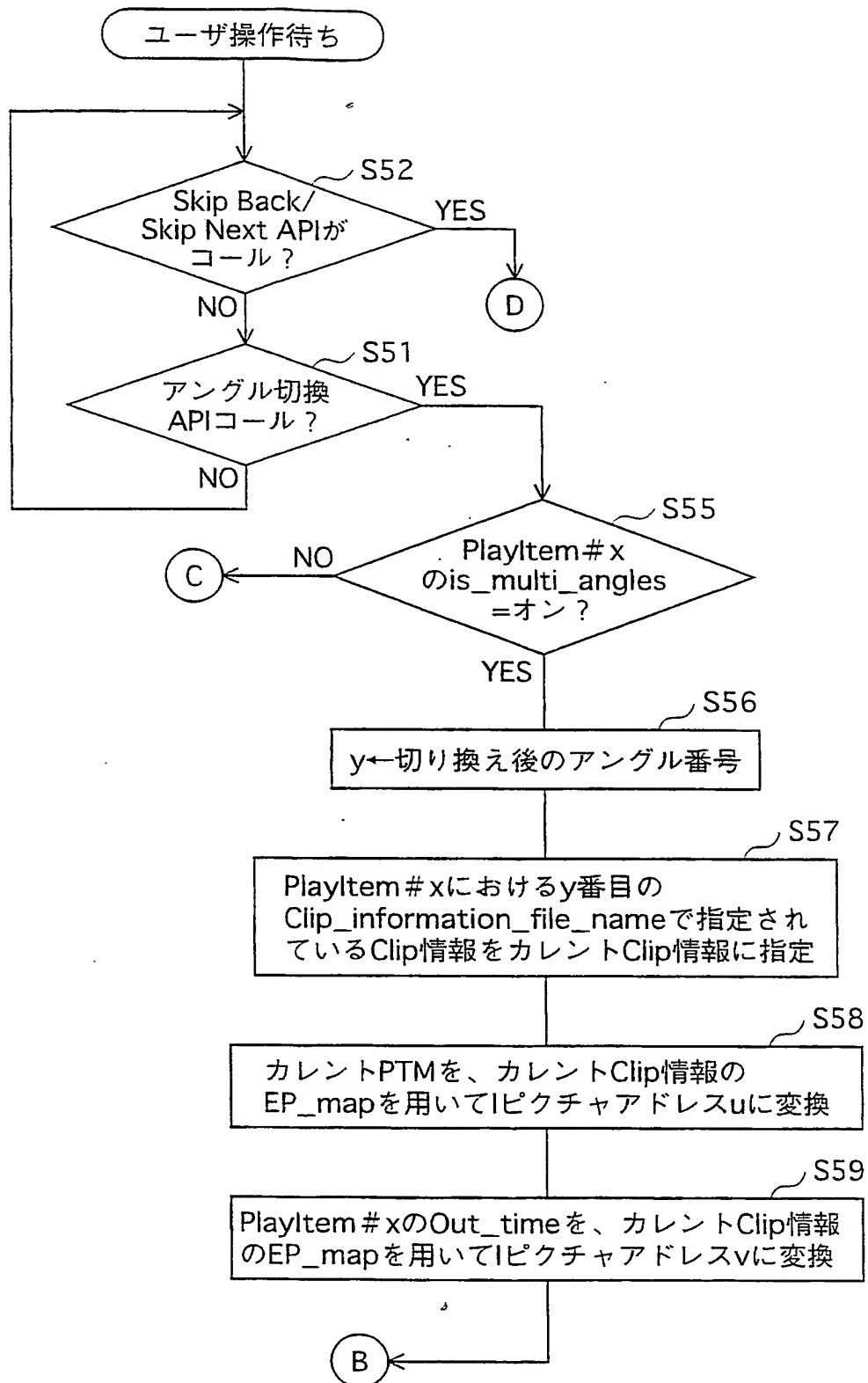


図35

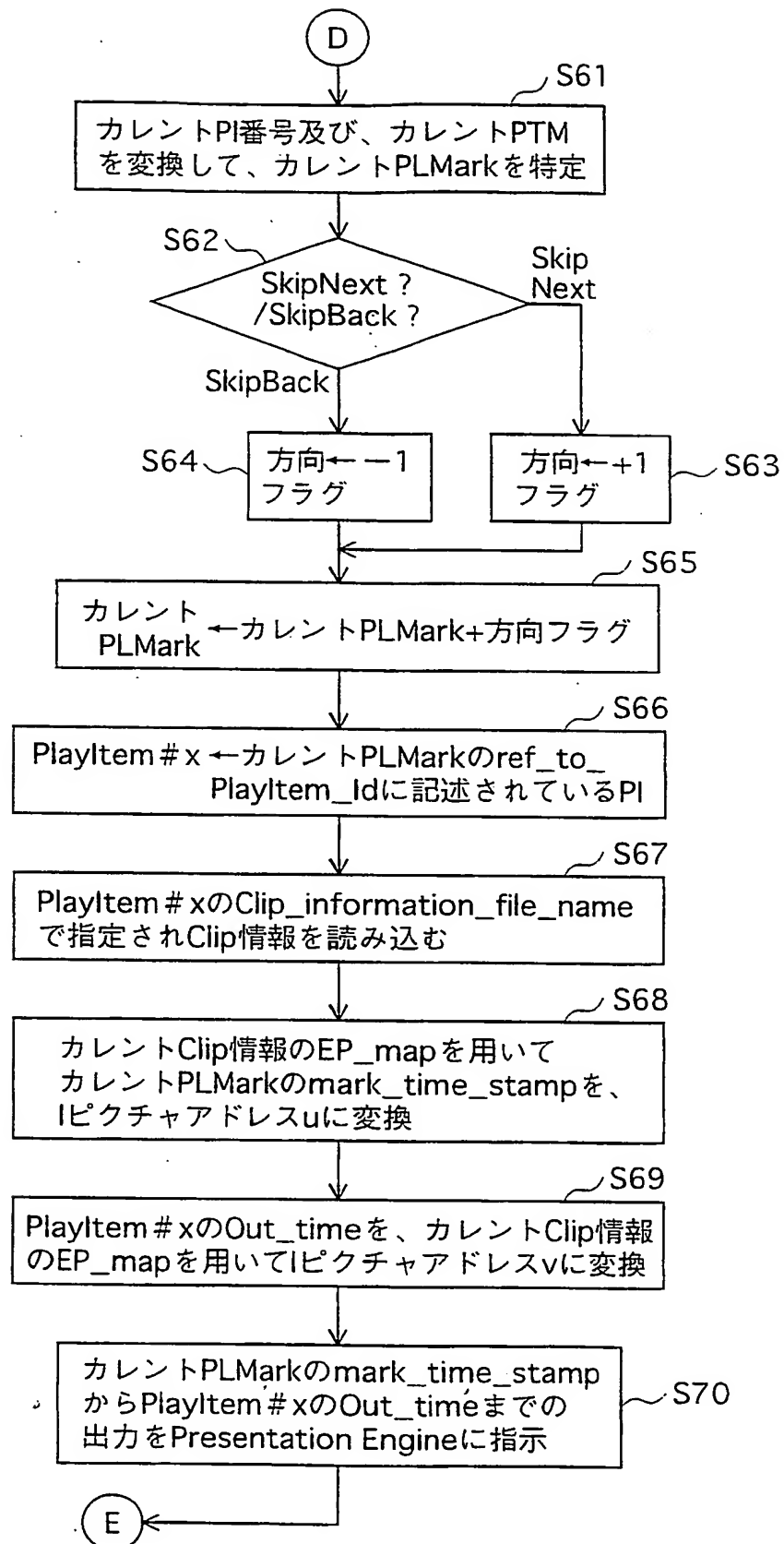


図36

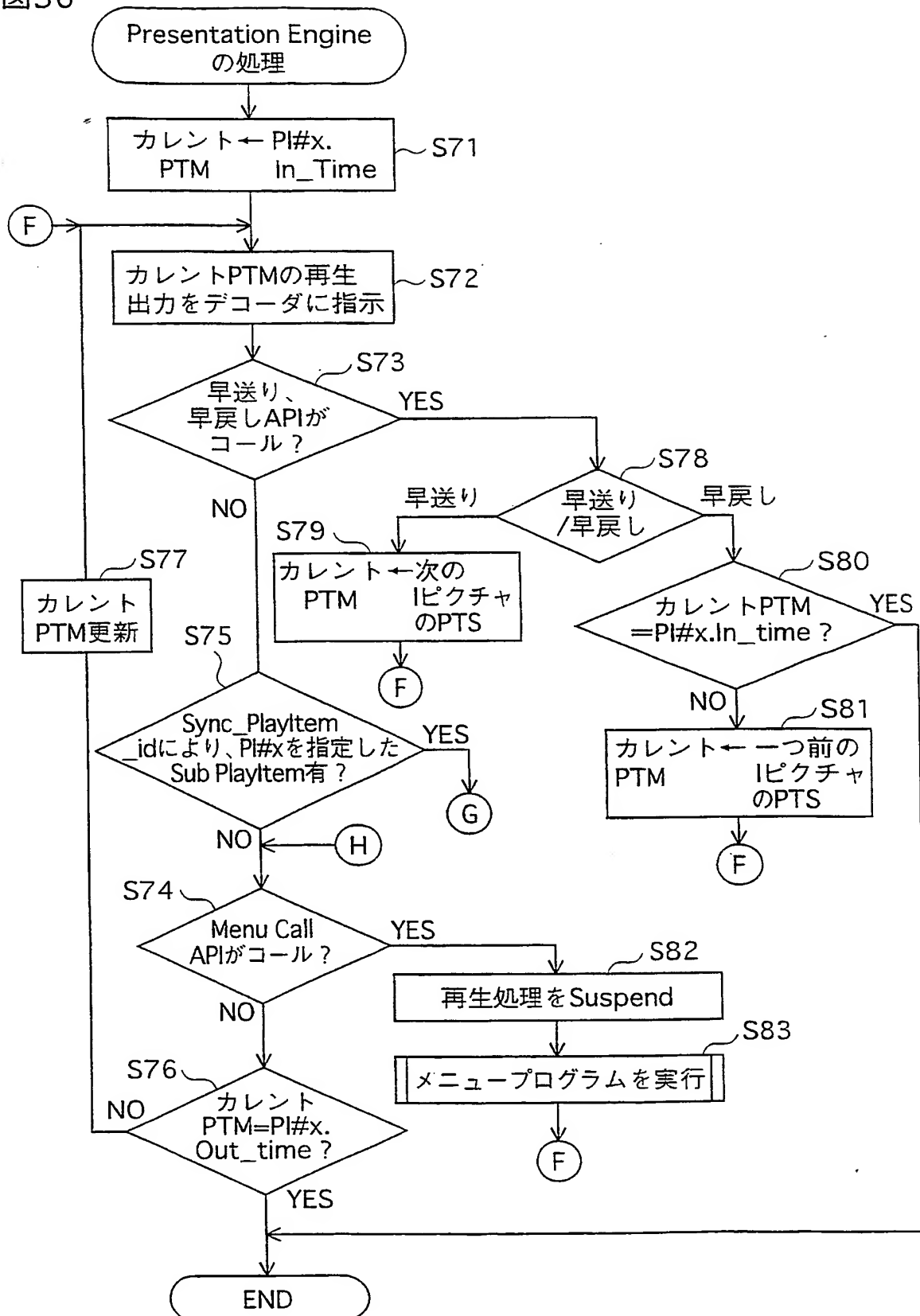


図37

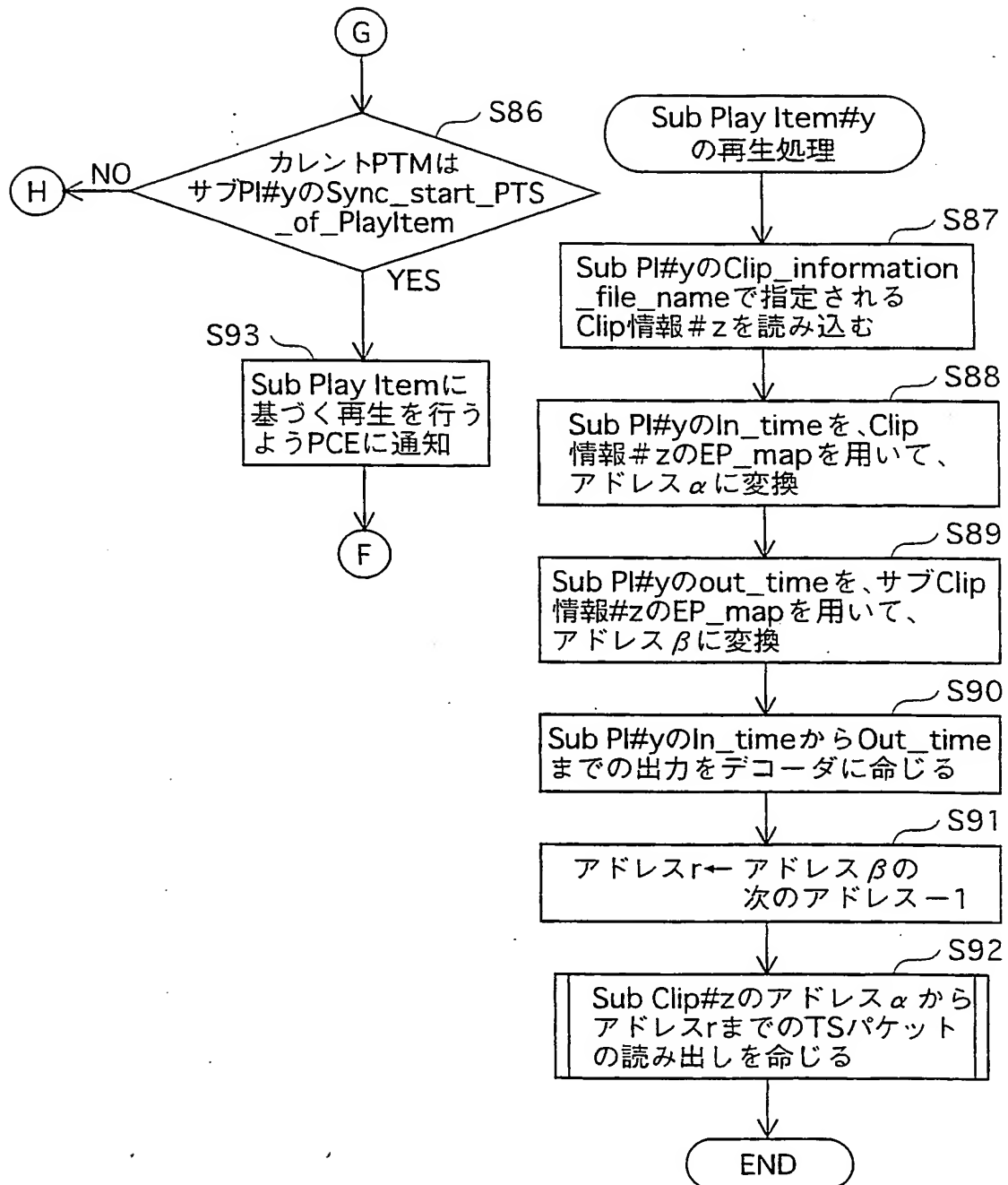


図38

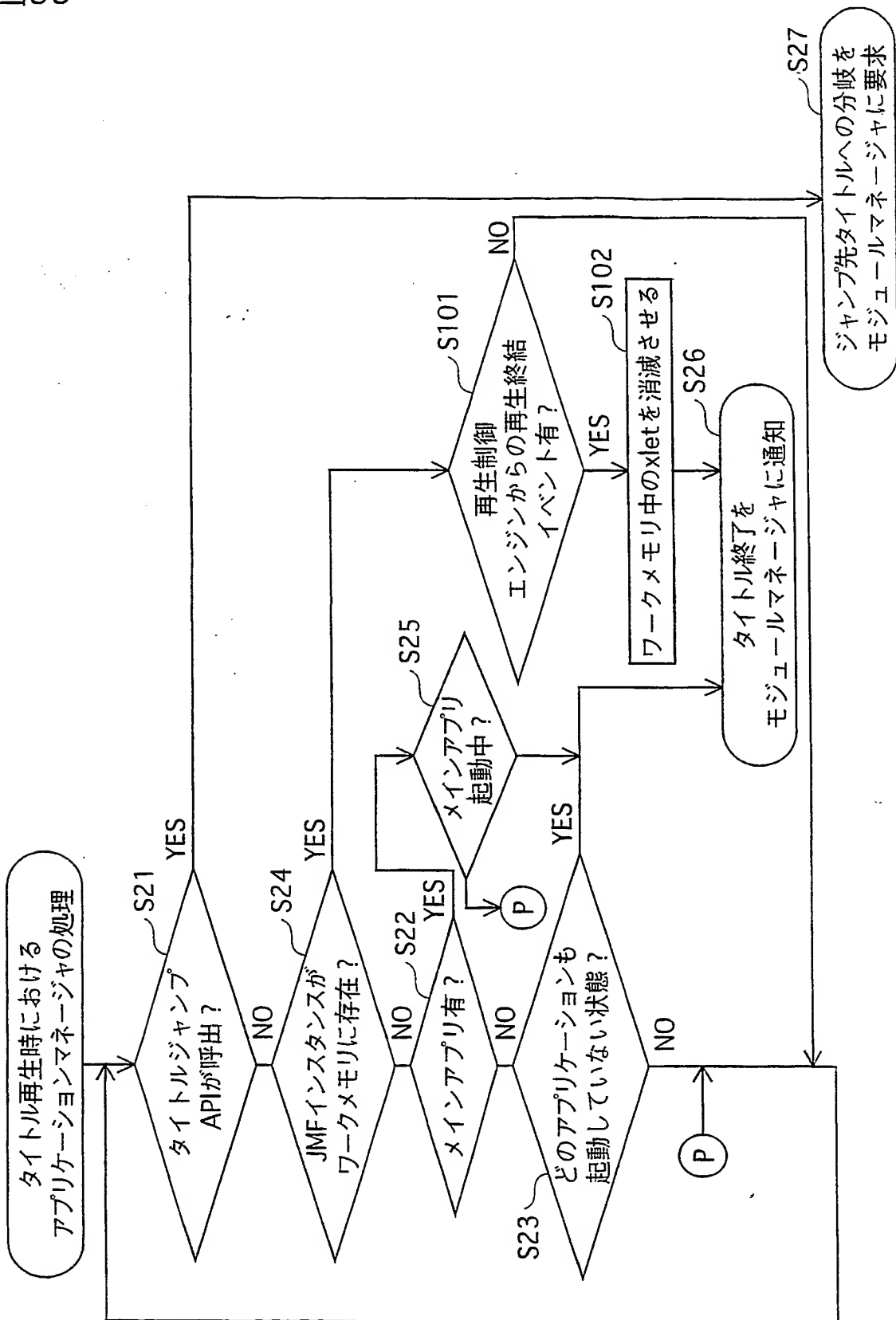


図39

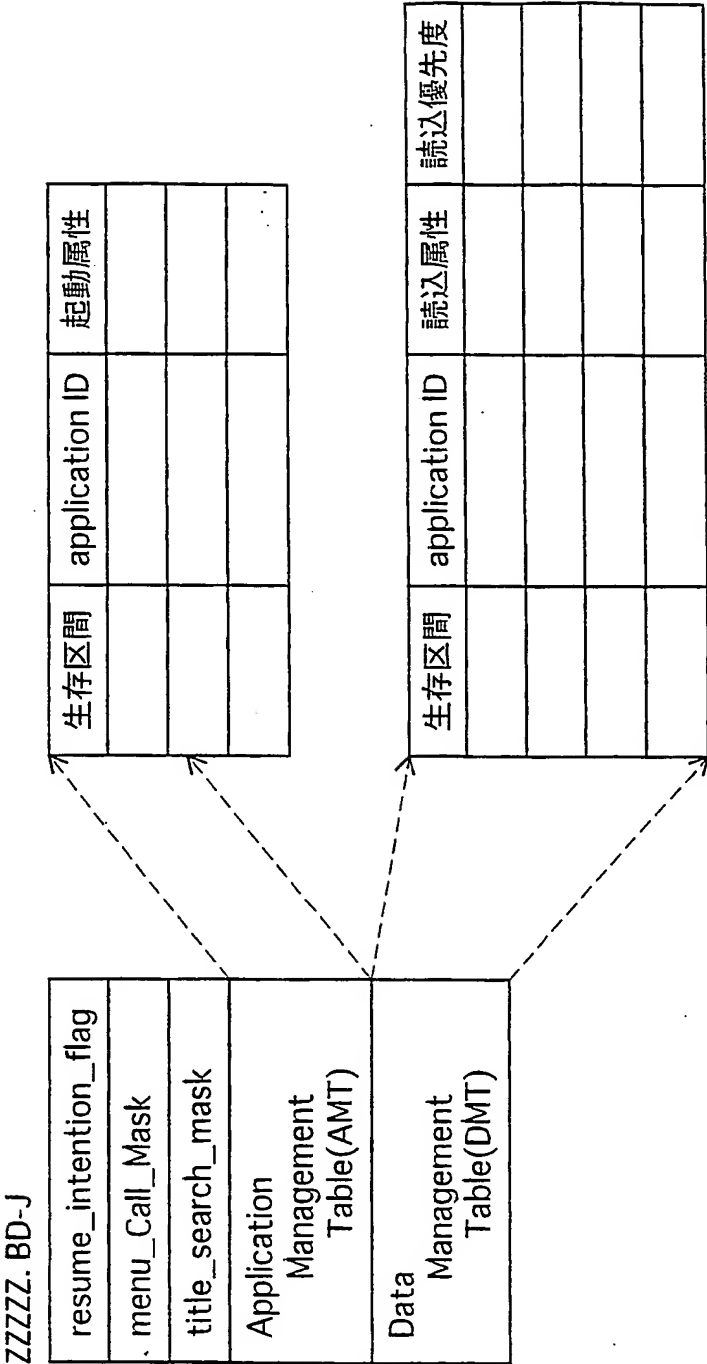


図40

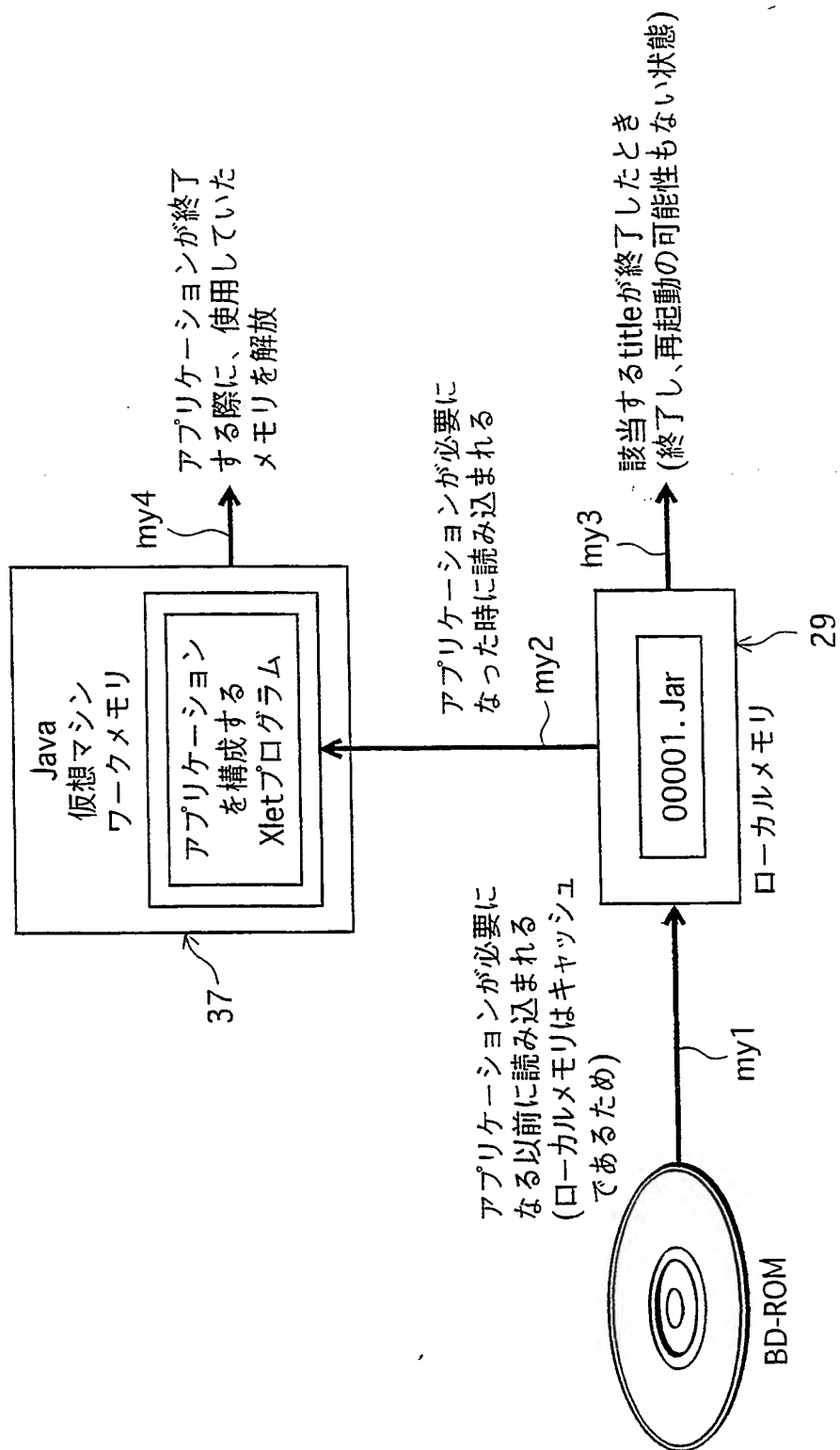


図41

(a)

title #1のデータ管理テーブル title #2のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度	生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1			title #1	application #1		
title #1	application #2			title #2	application #2		

(b)

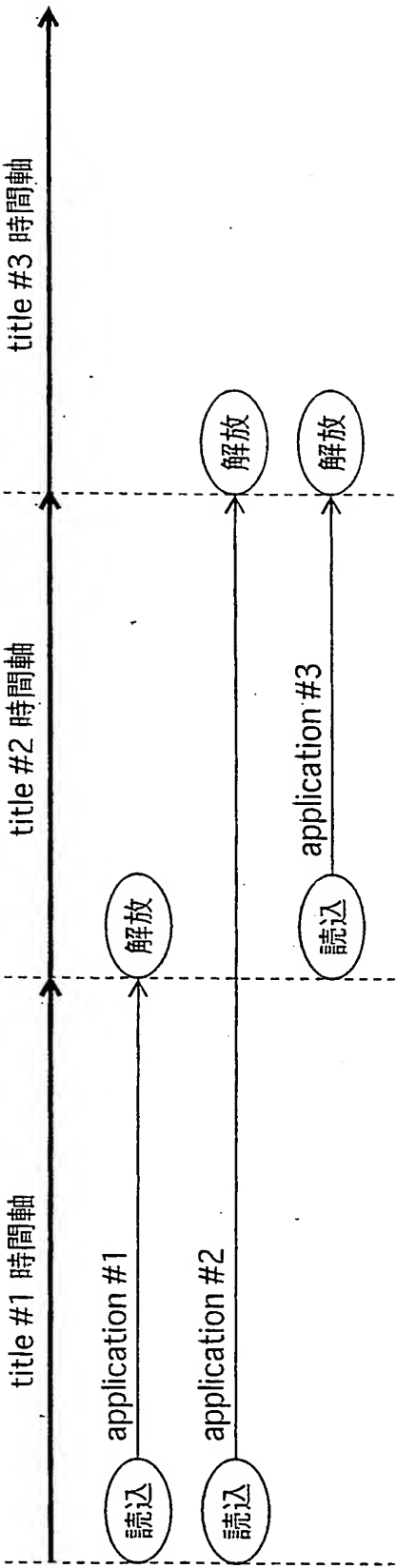


図42

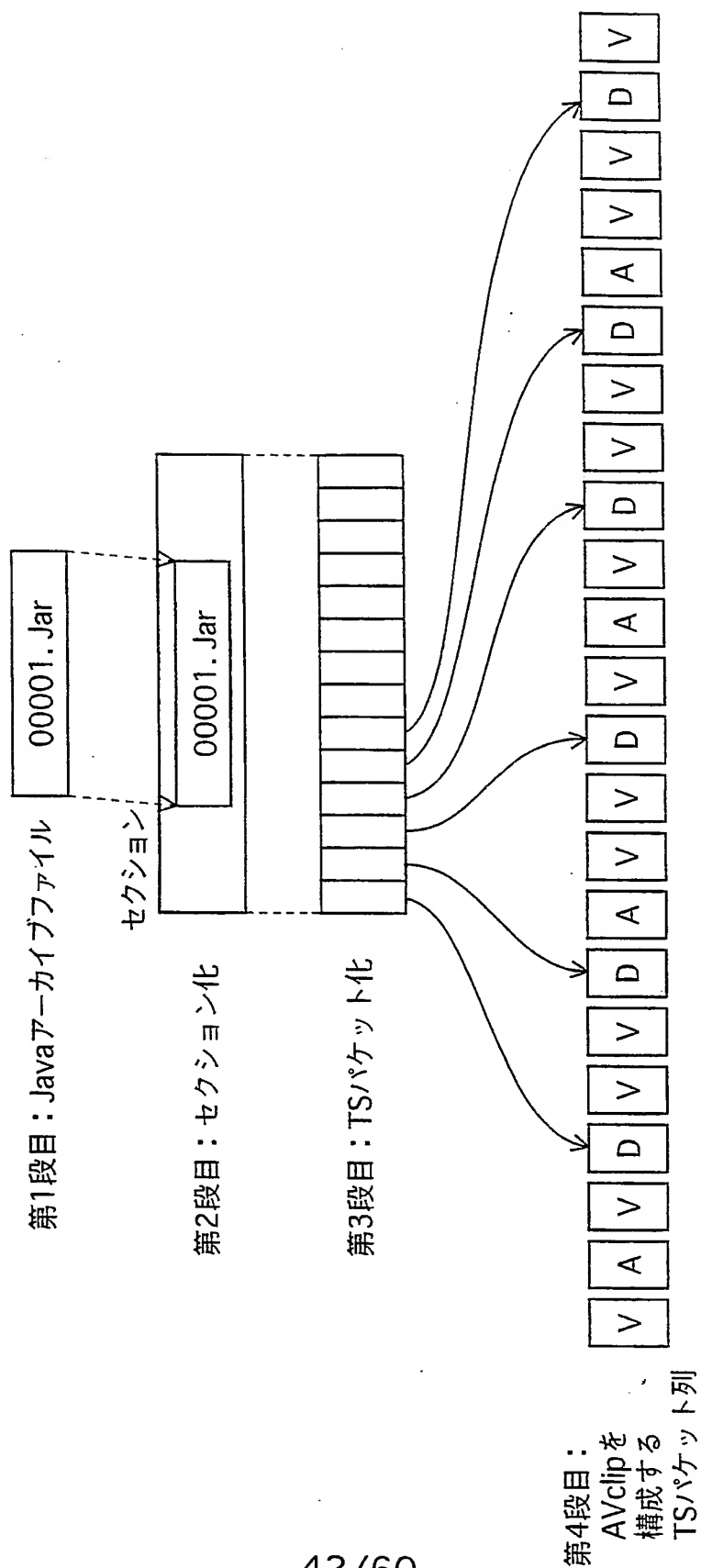


図43

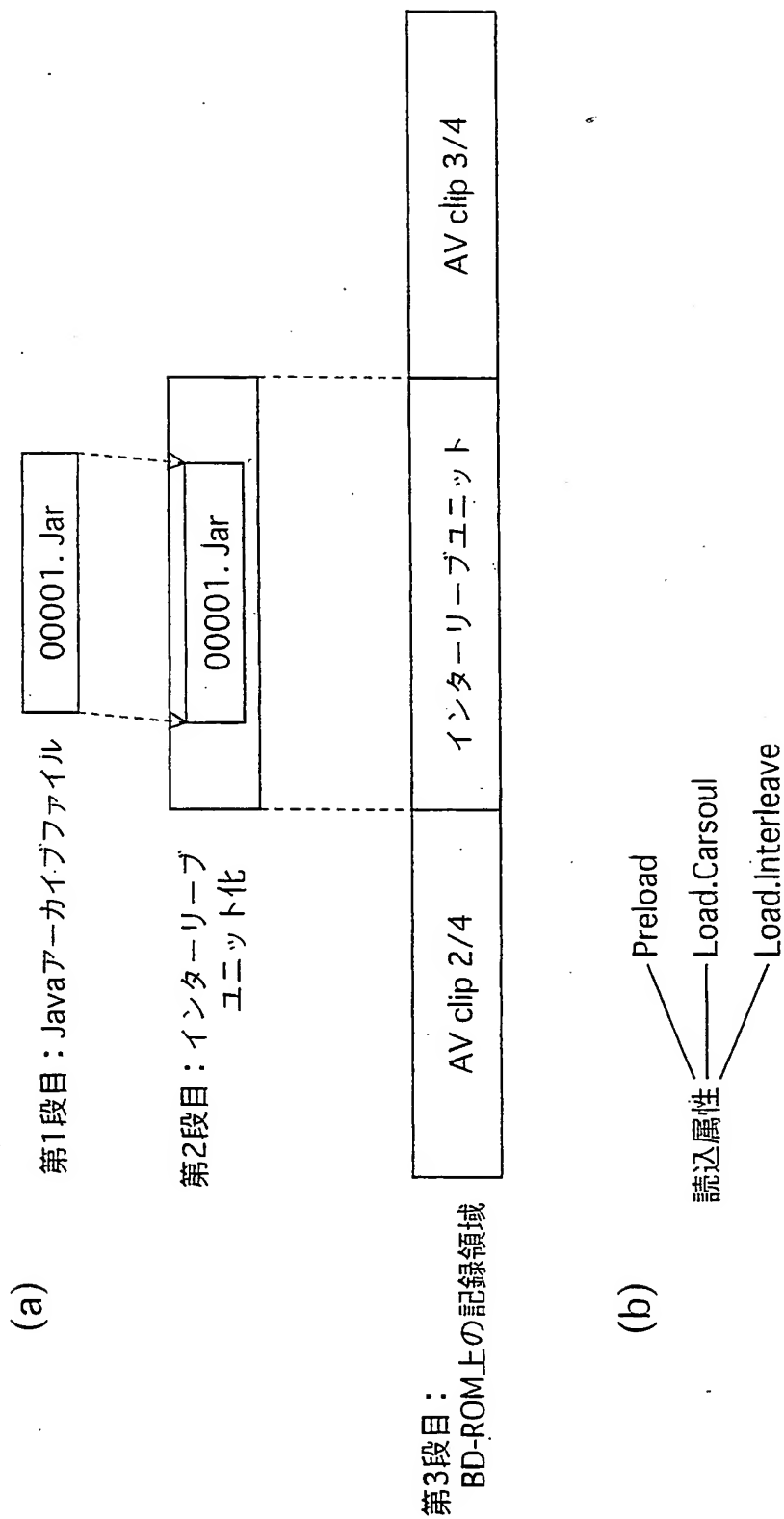


図44

(a) title #1のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1:chapter #1-#2	application #2	Load	optional
title #1:chapter #4-#5	application #3	Load	optional

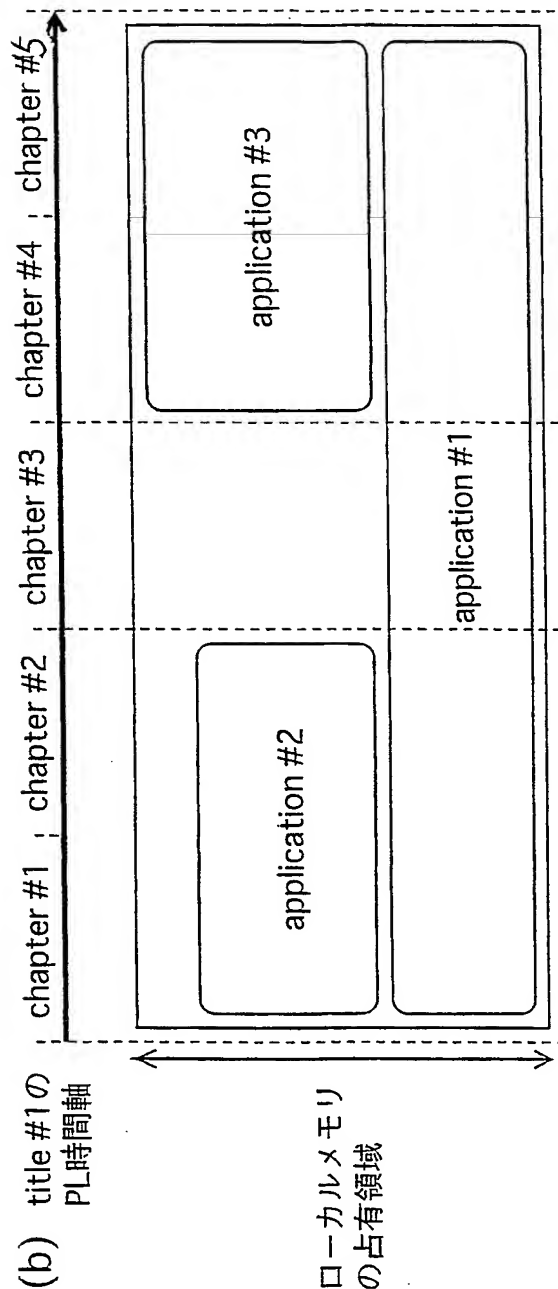
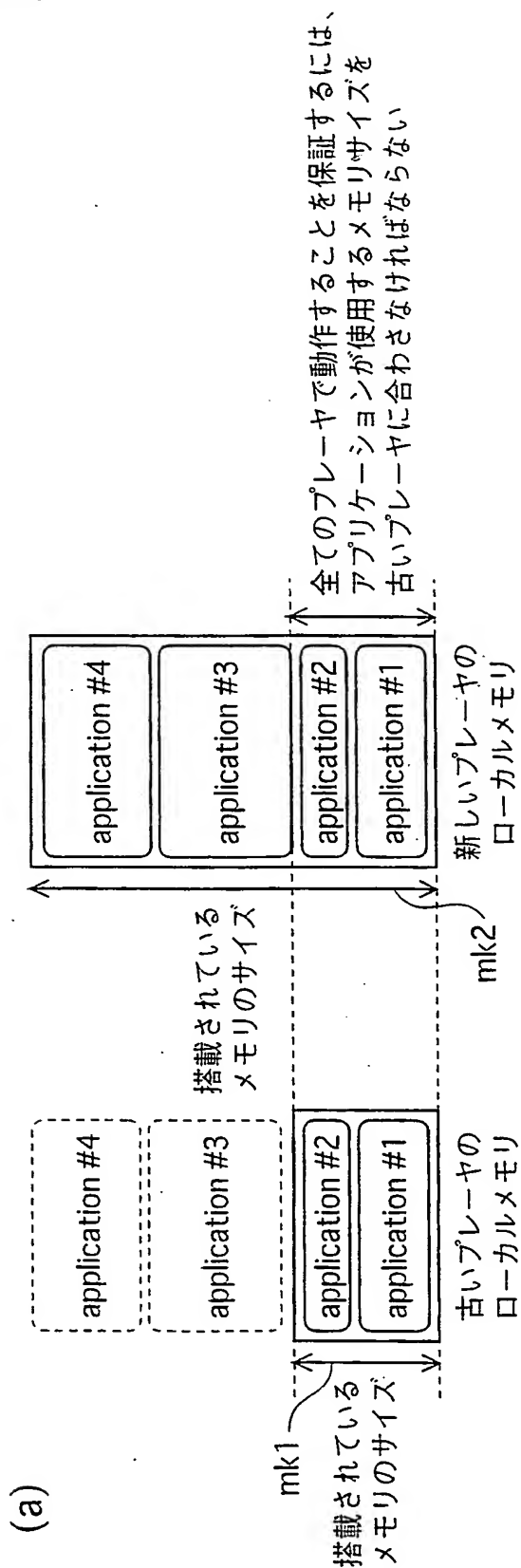


図45



(b) title #1のデータ管理テーブル

タイトル番号	アプリケーションID	読み属性	読み優先度
title #1	application #1	Preload	mandatory
title #1	application #2	Preload	mandatory
title #1	application #3	Preload	optional
title #1	application #4	Preload	optional

図46

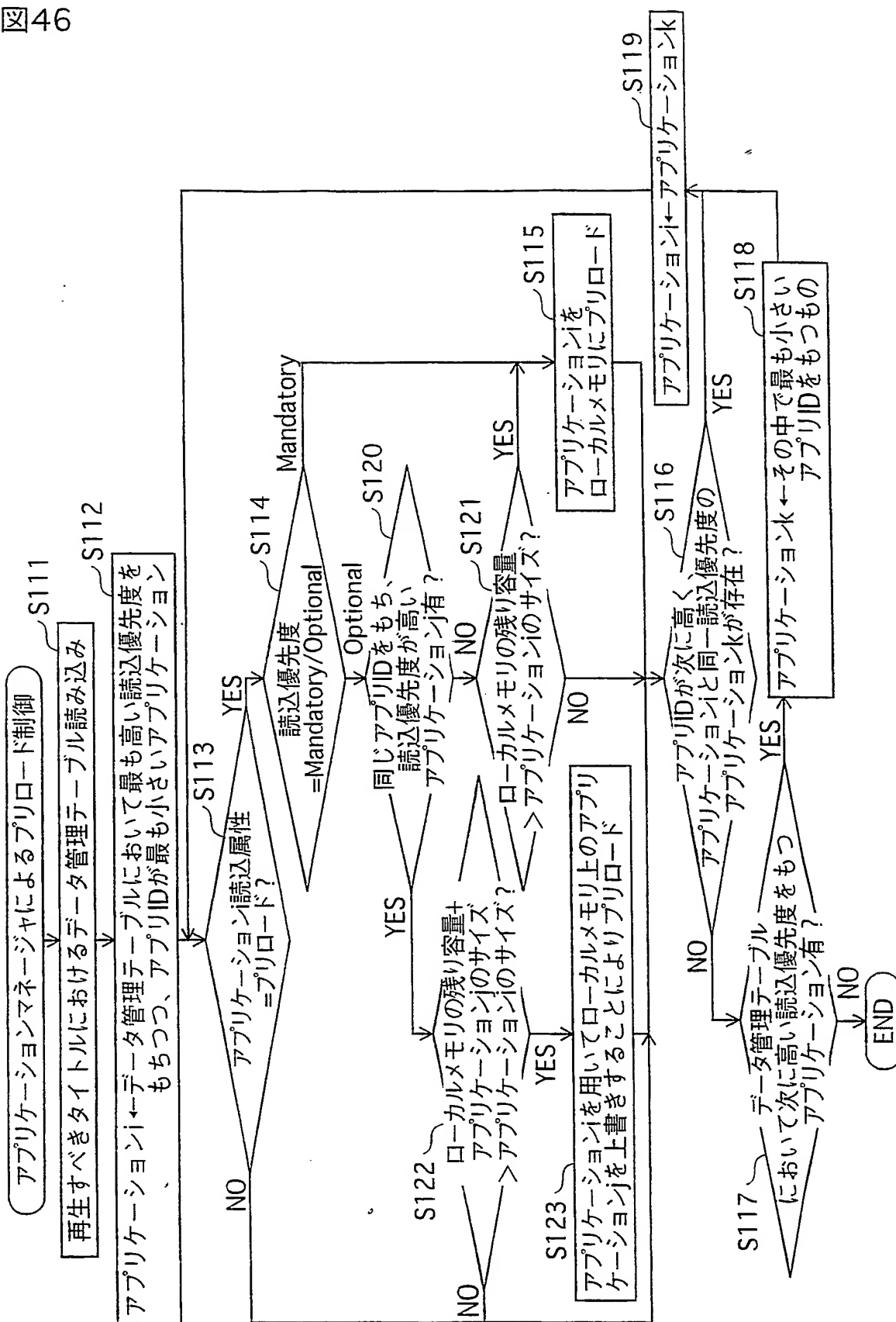


図47

(a) title #1のデータ管理テーブル

生存区間	アプリケーションID	読み属性	読み優先度
title #1	application #1	Preload	mandatory
title #1	application #1	Preload	optional:high
title #1	application #1	Preload	optional:low

(b) ローカルメモリの占有領域

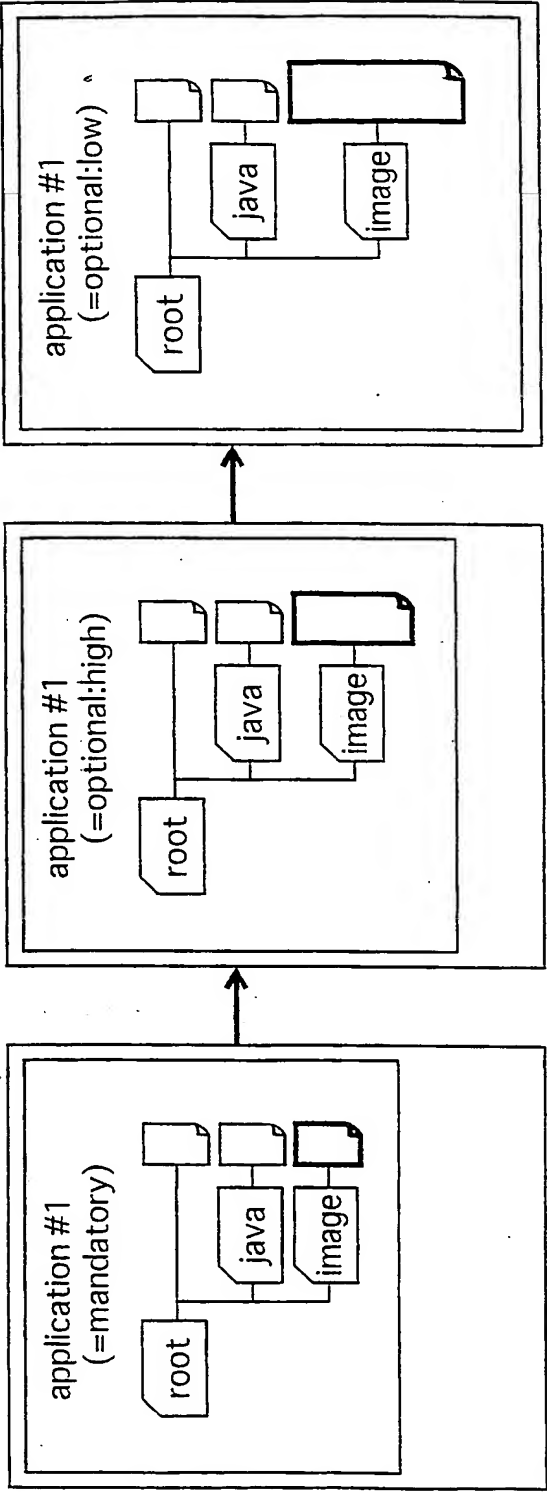


図48

(a) title #1のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1:chapter #1-#2	application #2	Load	mandatory
title #1:chapter #4-#5	application #3	Load	mandatory
title #1	application #3	Preload	optional

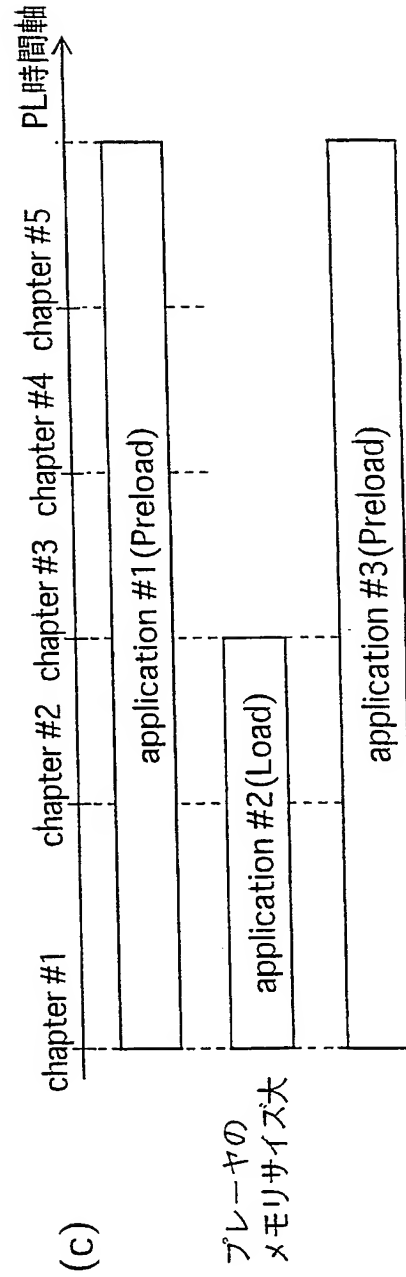
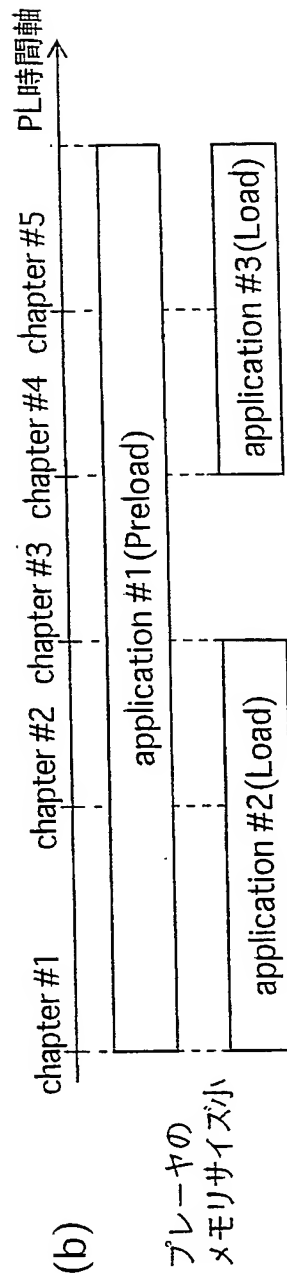


図49

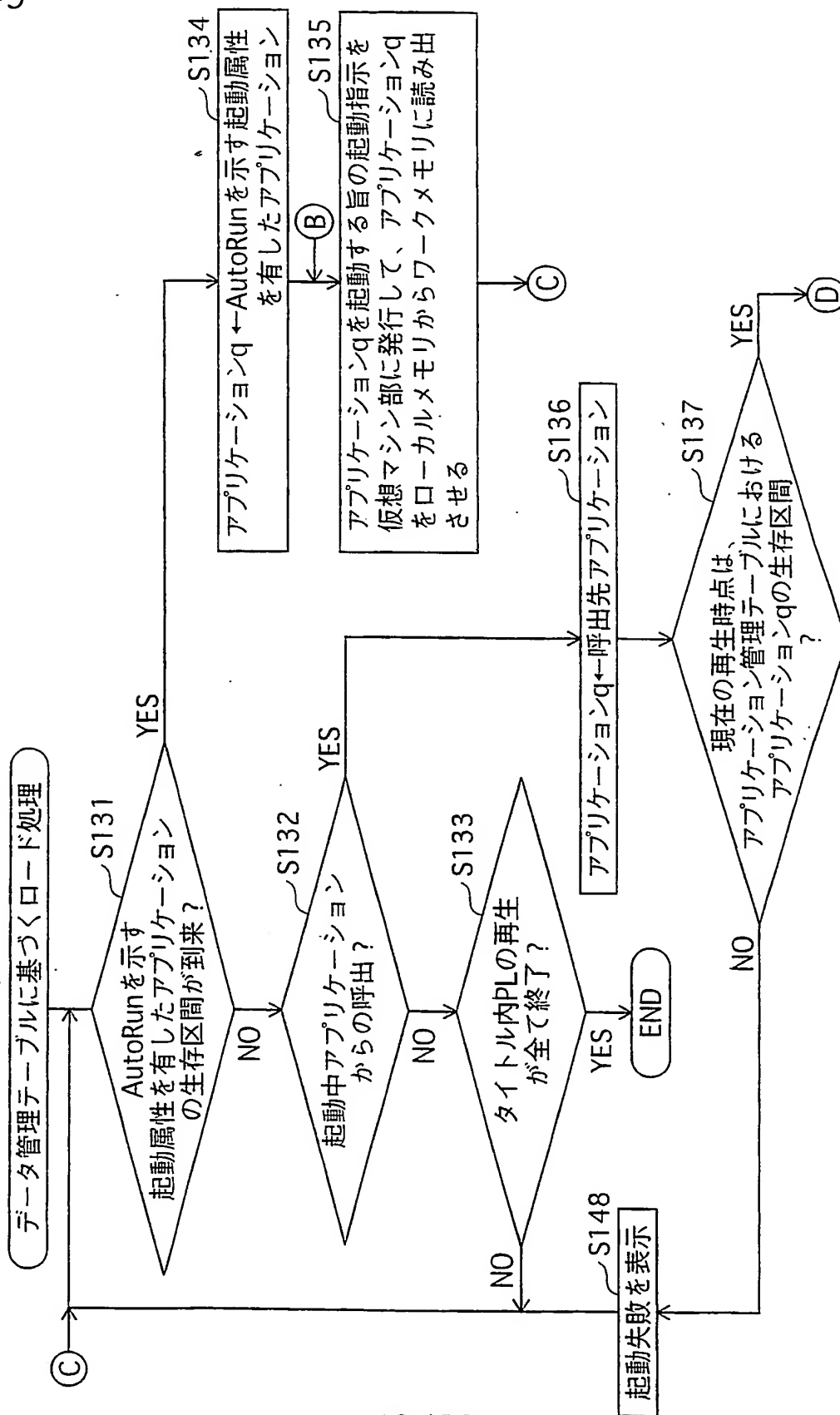


図50

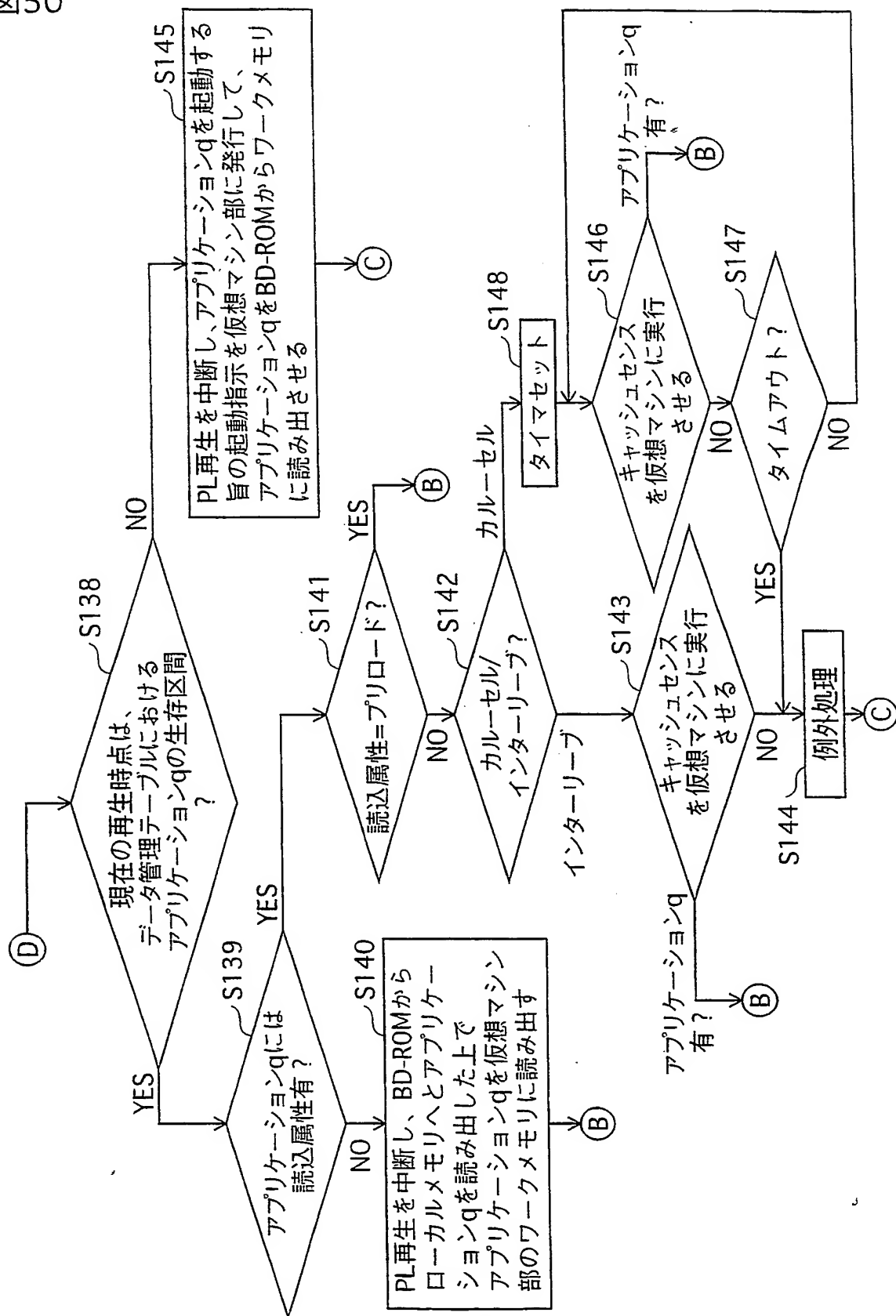


図51

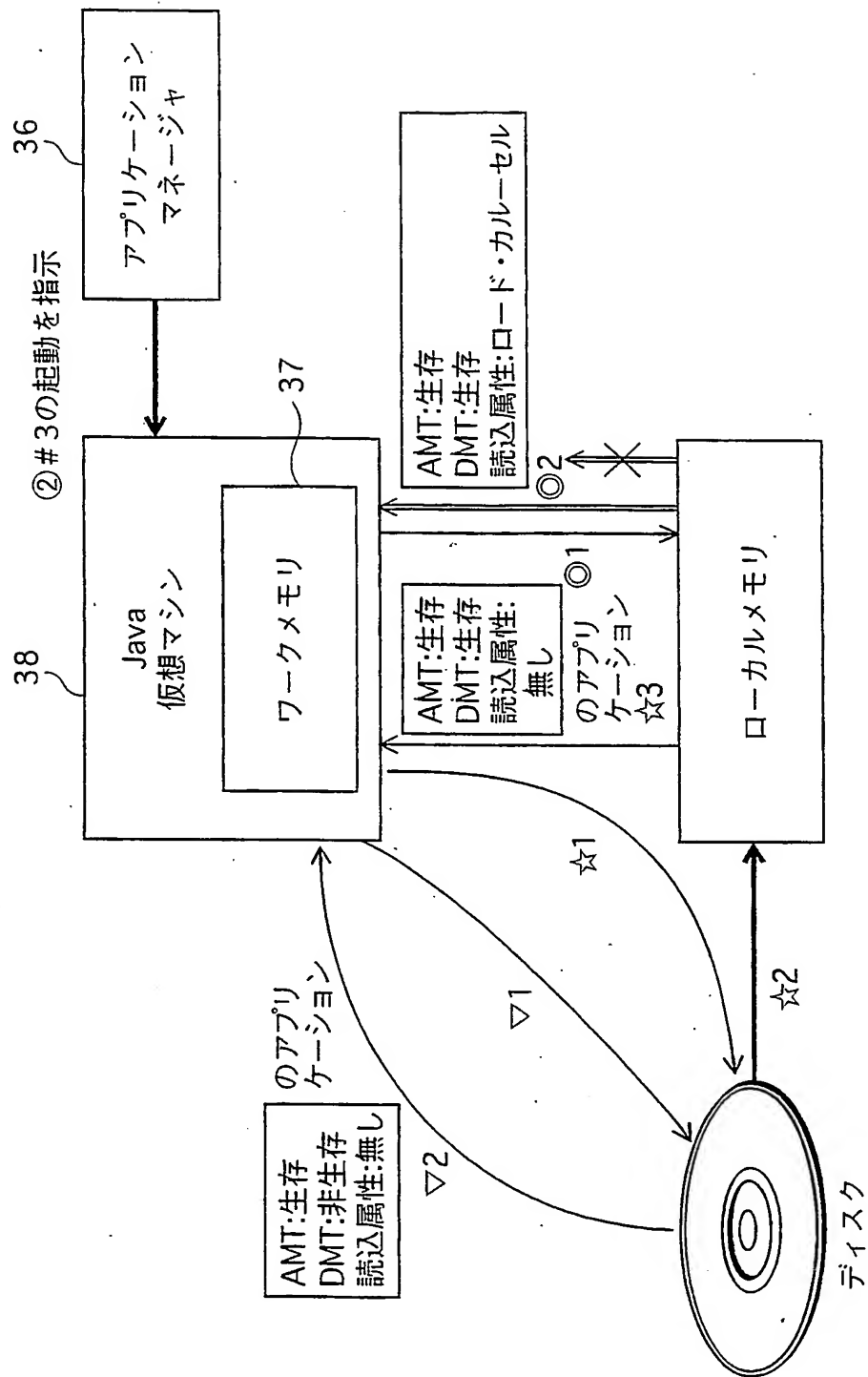


図52

(a) ZZZZZ. BD-J

resume_intention_flag
menu_Call_Mask
title_search_mask
Application Management Table(AMT)
Data Management Table(DMT)
PlayList Management Table(PLMT)

(b) PL管理テーブル

プレイリストID	再生属性
--	Auto Play
--	---

(c)

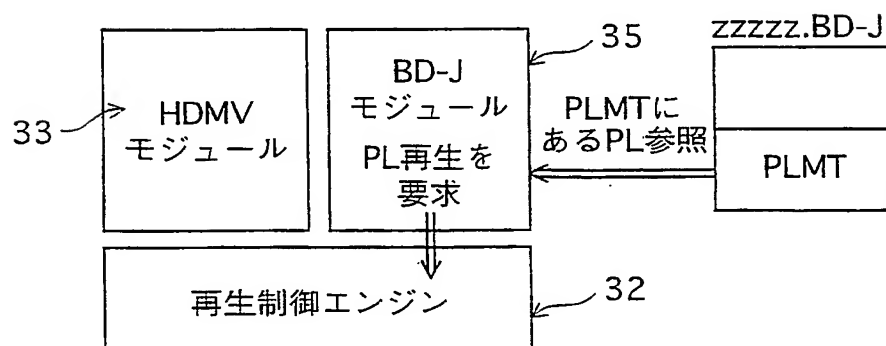


図53

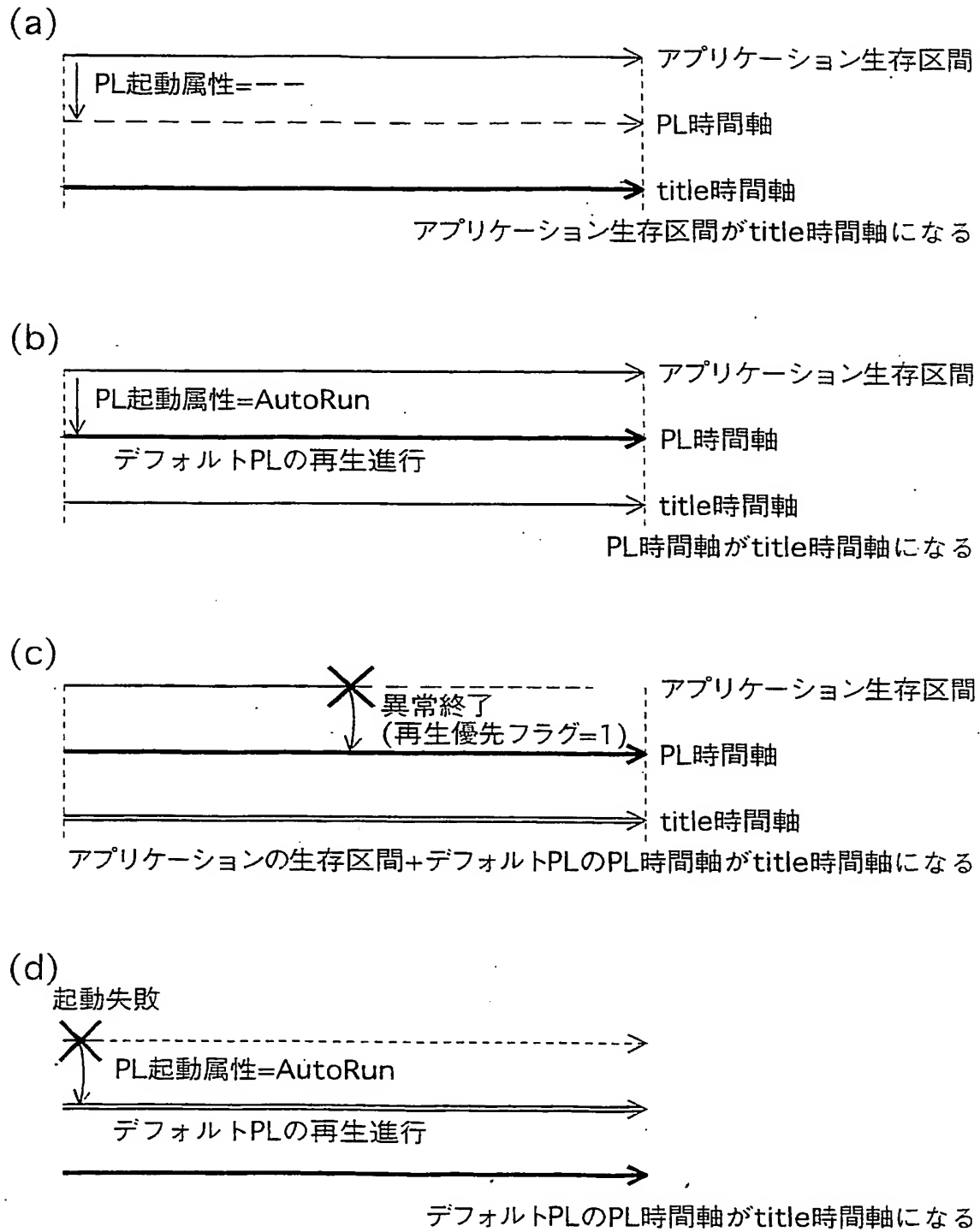


図54

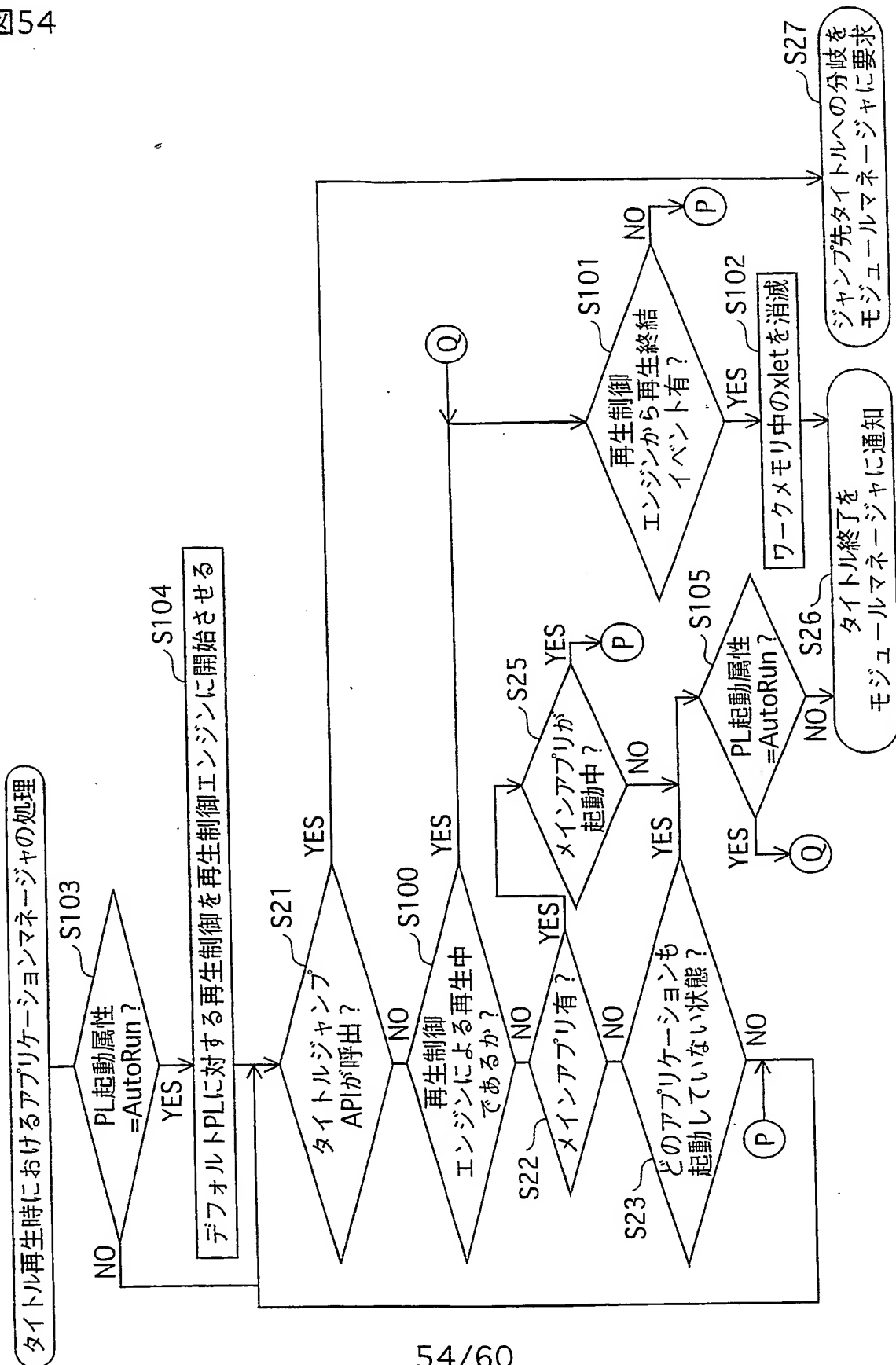


図55

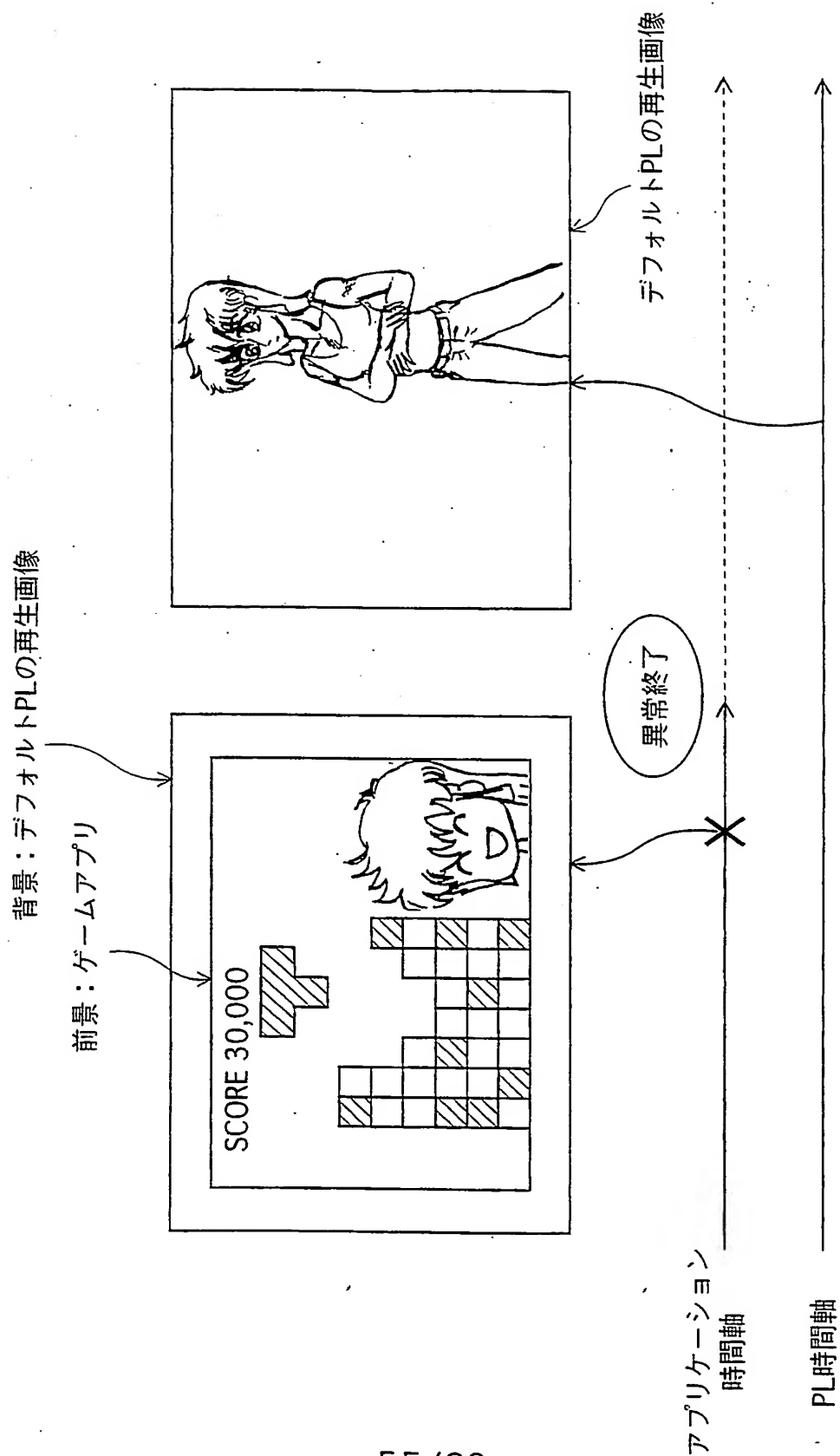


図56

(a) アプリケーション・データ管理テーブル

生存区間	アプリケーションID	起動属性	読込優先度
title #1	application #1	AutoRun	
title #1:chapter #1-#3	application #2	Ready	
title #1	application #3	--	
title #1:chapter #2-#4	application #4	Ready	

起動属性+読込属性

(b)

起動属性	AVストリーム再生 前のプリロード	自動起動 /呼出起動	ローカルメモリ へのロード	生存/非生存
AutoRun	○	自動	×	生存
AutoRun	×	自動	○	生存
READY	○	呼出	×	生存
READY	×	呼出	○	生存
無指定	×	呼出	×	生存

図57

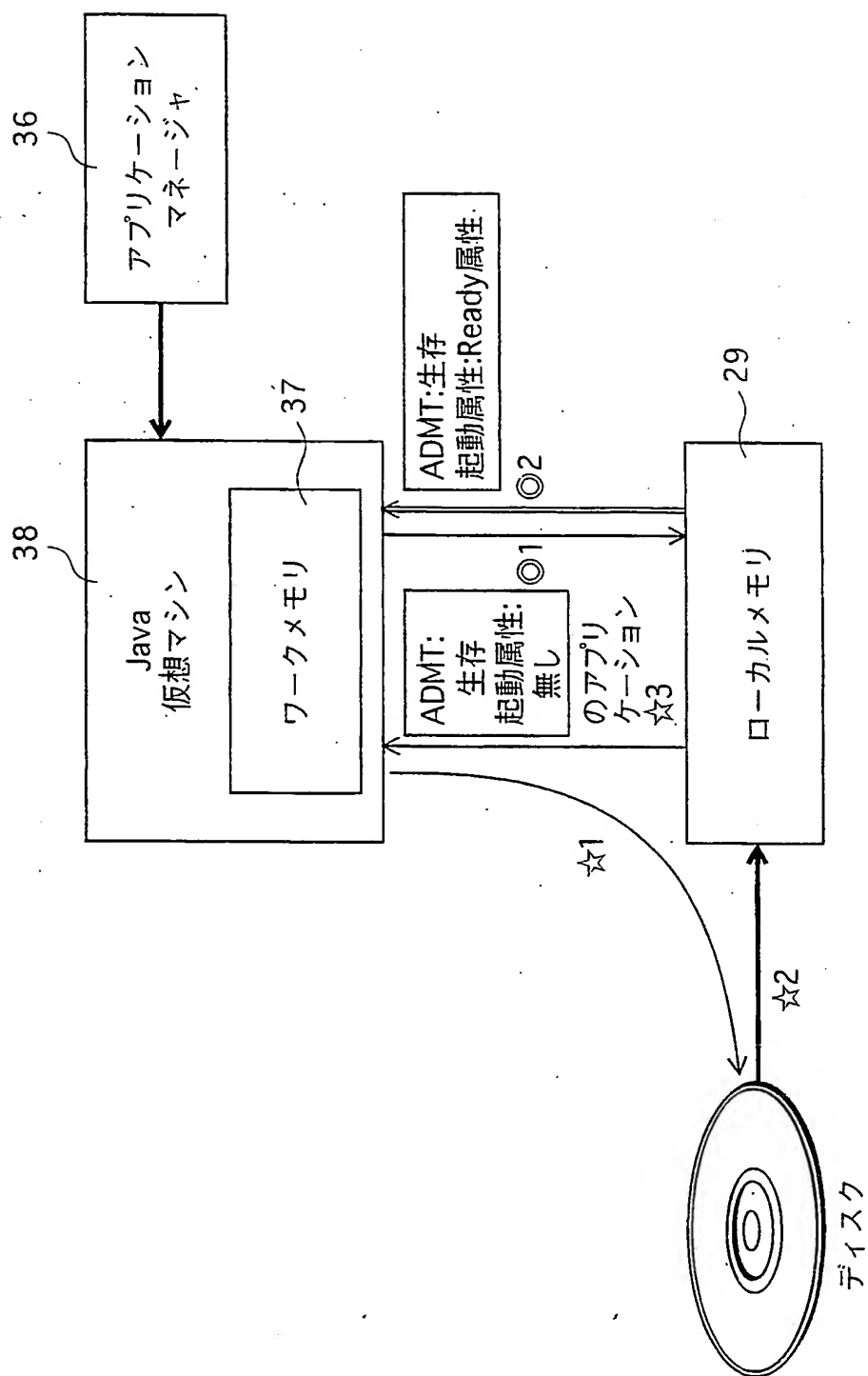


図58

(a) データ管理情報

生存区間	application ID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1	application #2	Preload	option, 255
title #1	application #3	Preload	option, 128

(b)

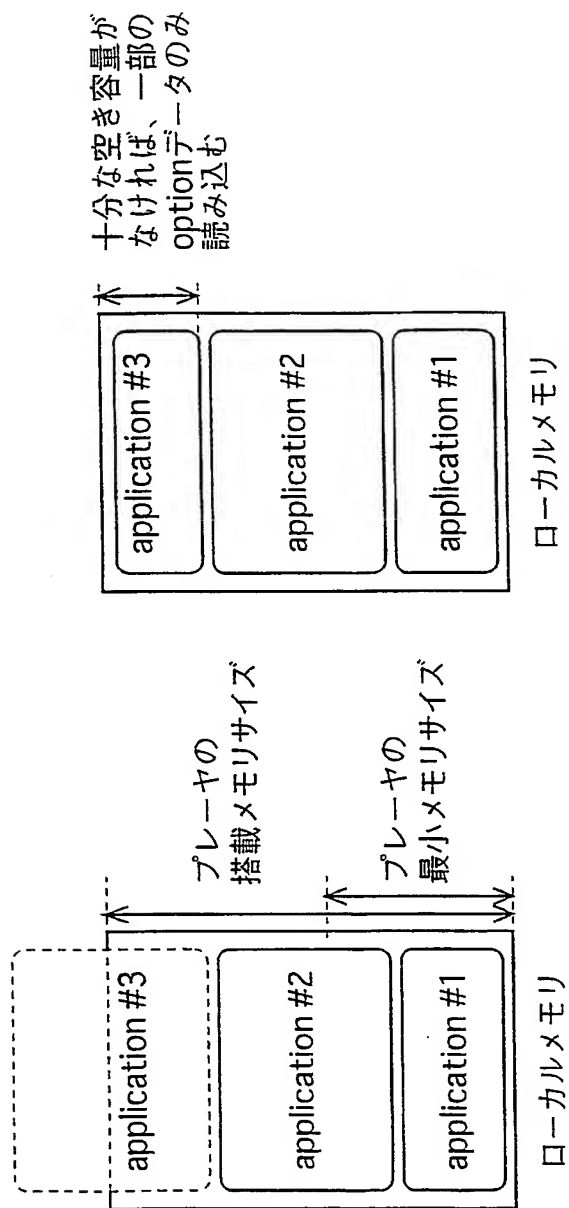


図59

(a) データ管理テーブル

生存区間	アプリケーションID	ロード属性	読込優先度	グループ属性
title #1	application #1		mandatory	—
title #1	application #2		optional	group #1
title #1	application #3		optional	group #1

(b)

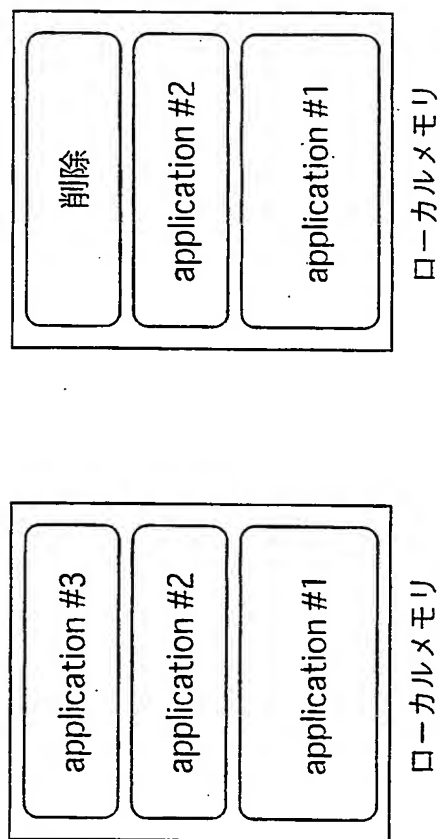
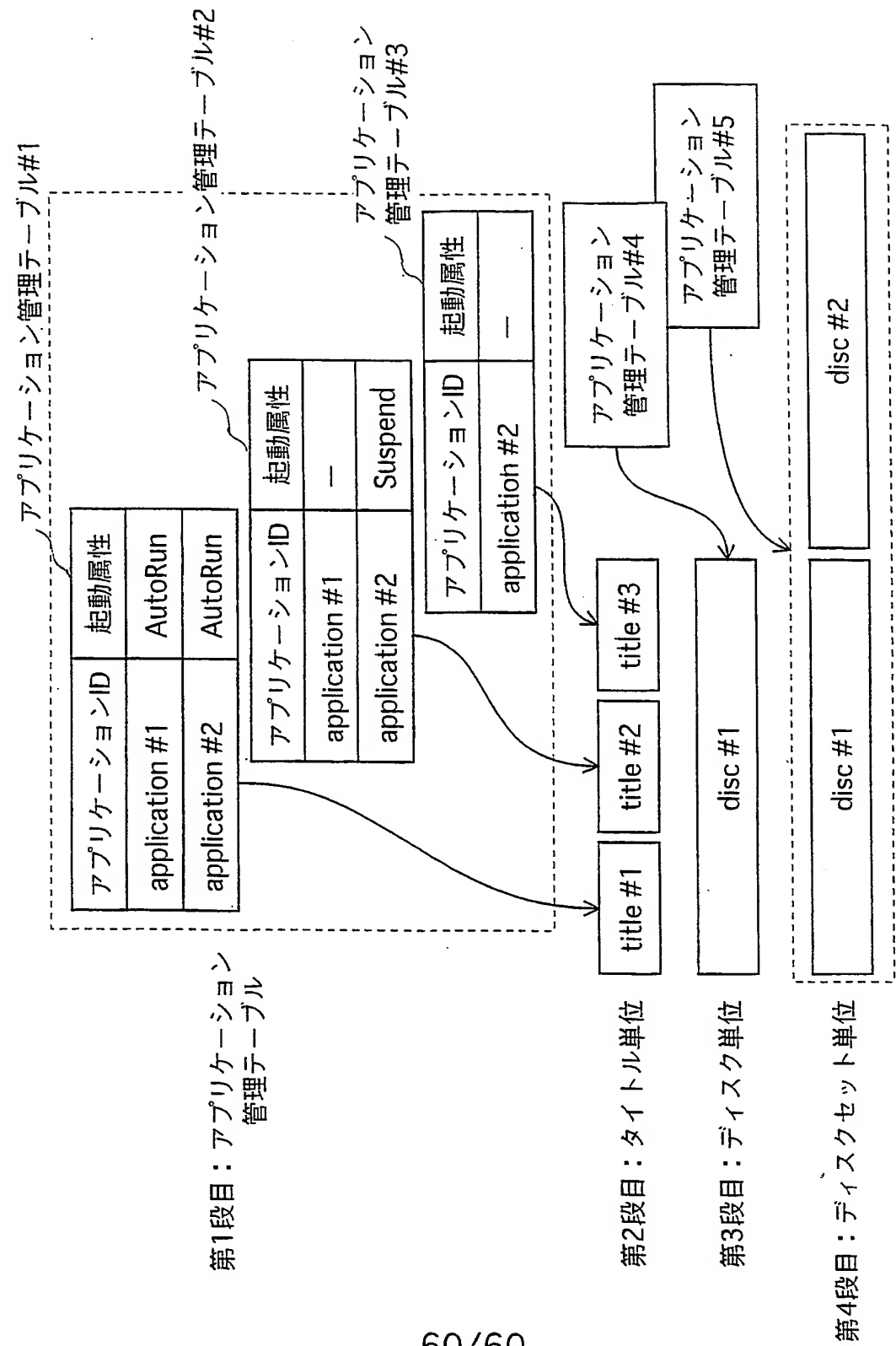


図60



INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2004/015335

A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl⁷ G11B20/10, G11B20/12, G11B27/00, G11B27/10, G06F19/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl⁷ G11B20/10, G11B20/12, G11B27/00, G11B27/10, G06F19/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2005
Kokai Jitsuyo Shinan Koho	1971-2005	Jitsuyo Shinan Toroku Koho	1996-2005

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2002-369154 A (Matsushita Electric Industrial Co., Ltd.), 20 December, 2002 (20.12.02), Full text; Figs. 1 to 39 & WO 02/082810 A1	1-7
A	WO 00/078043 A (WINK COMMUNICATIONS, INC.), 21 December, 2000 (21.12.00), Claim 3; Fig. 1 & WO 00/078043 A1	1-2
A	JP 2003-230104 A (Matsushita Electric Industrial Co., Ltd.), 15 August, 2003 (15.08.03), Par. No. [0004]; Fig. 1 & WO 03/047244 A1	5

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure; use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search
06 January, 2005 (06.01.05)

Date of mailing of the international search report
25 January, 2005 (25.01.05)

Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

国際調査報告

国際出願番号 PCT/J P 2004/015335

A. 発明の属する分野の分類 (国際特許分類 (IPC))

Int. Cl⁷ G11B 20/10 G11B 20/12 G11B 27/00 G11B 27/10
G06F 19/00

B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))

Int. Cl⁷ G11B 20/10 G11B 20/12 G11B 27/00 G11B 27/10
G06F 19/00

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1922-1996年
 日本国公開実用新案公報 1971-2005年
 日本国登録実用新案公報 1994-2005年
 日本国実用新案登録公報 1996-2005年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
A	J P 2002-369154 A (松下電器産業株式会社) 2002. 12. 20 , 全文, 第1-39図 & WO 02/082810 A 1	1-7
A	WO 00/078043 A (ウィンク・コミュニケーションズ・イン コーポレーテッド) 2000. 12. 21 , 【請求項3】 , 第1図 & WO 00/078043 A 1	1-2
A	J P 2003-230104 A (松下電器産業株式会社) 2003. 08. 15 , 【0004】 , 第1図 & WO 03/047244 A 1	5

☐ C欄の続きにも文献が列挙されている。☐ パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」 特に関連のある文献ではなく、一般的技術水準を示すもの
 「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
 「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)
 「O」 口頭による開示、使用、展示等に言及する文献
 「P」 国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
 「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
 「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
 「&」 同一パテントファミリー文献

国際調査を完了した日

06. 01. 2005

国際調査報告の発送日

25. 1. 2005

国際調査機関の名称及びあて先

日本国特許庁 (ISA/J P)
 郵便番号100-8915
 東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)

前田 祐希

5 Q

2946

電話番号 03-3581-1101 内線 3590